Lecture Notes in Artificial Intelligence 1415

Subseries of Lecture Notes in Computer Science

Edited by J. G. Carbonell and J. Siekmann

Lecture Notes in Computer Science

Edited by G. Goos, J. Hartmanis and J. van Leeuwen

Springer
Berlin
Heidelberg
New York
Barcelona
Budapest
Hong Kong
London
Milan
Paris
Santa Clara
Singapore
Tokyo

José Mira   Angel Pasqual del Pobil
Moonis Ali (Eds.)

Methodology and Tools in Knowledge-Based Systems

11th International Conference
on Industrial and Engineering Applications
of Artificial Intelligence and Expert Systems
IEA-98-AIE
Benicàssim, Castellón, Spain, June 1-4, 1998
Proceedings, Volume I

Springer

Volume Editors

Angel Pasqual del Pobil
Department of Informatics, Jaume-I University
Campus Penyeta Roja, E-12071 Castellón, Spain
E-mail: pobil@inf.uji.es

José Mira
Departamento de InteligenciaArtificial, Facultad de Ciencias
Universidad Nacinoal de Educación a Distancia
Sende del Rey, s/n, E-28040 Madrid, Spain
E-mail: jmira@dia.uned.es

Moonis Ali
Department of Computer Science, SouthwestTexas State University
San Marcos, TX 78666-4616, USA
E-mail: ma04@swt.edu

Cataloging-in-Publication Data applied for

Die Deutsche Bibliothek - CIP-Einheitsaufnahme

International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems <11, 1998, Benicasim; Castellón de la Plana>:
11th International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems : Benicàssim, Castellón, Spain, June 1 - 4, 1998 ; proceedings / IEA-98-AIE. - Berlin ; Heidelberg ; New York ; Barcelona ; Budapest ; Hong Kong ; London ; Milan ; Paris ; Santa Clara ; Singapore ; Tokyo : Springer
Vol. 1. Methodology and tools in knowledge based systems / José Mira ... (ed.). - 1998
(Lecture notes in computer science ; Vol. 1415 : Lecture notes in artificial intelligence)
ISBN 3-540-64582-9

CR Subject Classification (1991): I.2, J.2, J.6

ISBN 3-540-64582-9 Springer-Verlag Berlin Heidelberg New York

Typesetting: Camera ready by author
SPIN 10637249 06/3142 – 5 4 3 2 1 0 Printed on acid-free paper

Preface

Nowadays, it is generally accepted that the aim of Applied Artificial Intelligence is to render computational a large portion of non-analytical human knowledge. To attain this end, we need first to build knowledge-level models of analysis and synthesis tasks in scientific and technical domains, such as those performed daily by human experts in fields such as medical diagnosis, design in civil or telecommunication engineering, architecture, flexible manufacturing, or tutoring. Then, these models have to be transformed in such a way that their entities and relations can be linked to the primitives of a programming language and, finally, produce a program and continue with the usual phases in software engineering (validation, evaluation, and maintenance).

This purpose, that seems to be clear, has suffered since its origins in 1956, from a lack of methodology and foundations. That is, there has been an excessive hurry to develop applications (*expert* systems) without the technical and methodological support available to other engineering disciplines —those dealing with matter or energy— having been established. This is the reason why the advancement of Knowledge Engineering has not been as robust as expected.

Fortunately, interest in methodology and foundations has grown in recent years, commencing by Clancey and Chandrasekaran's proposals about generic tasks aiming at capturing recurrent abstractions in human knowledge modeling. Then, efforts have been made to build libraries of problem-solving methods to develop these tasks by decomposing them up to primitive level and completing these tasks and methods with ontologies and domain knowledge models together with a set of assumptions about implicit representations for each method and about the method's assumptions which are implicit in each domain model. These three basic concepts —tasks, method, and domain—, along with the underlying pursuit of designing reusable components, have characterized most of methodological developments around KADS, CommonKADS, and PROTÉGÉ, for instance.

The scope and topics included in the Call for Papers of the *Eleventh International Conference on Industrial and Engineering Applications of Artificial Intelligence and Expert Systems* (IEA/AIE-98) were compiled within this spirit of concern about sound foundations and methodology, as well as with the explicit acknowledgment of the necessity of developing efficient procedures to make the models operational. As a result of this call, 291 contributed and invited papers were submitted from 41 countries; the program committee selected 187 among them, after conscientiously considering the reviews provided by at least two referees per paper. We believe that the significant increase in the number of submitted papers, with respect to recent conferences, is a symptom of a maturing interest within the AI community towards fundamental issues relevant to well-founded and robust applications in the real world.

We are pleased to present, as program chairs and editors of these two volumes, a final version of the accepted papers incorporating the reviewers' comments. We have arranged their contents basically following the topic list included in the Call for Papers, adding some additional topics which received special attention as a result of being the subject of invited sessions. The first volume entitled *Methodology and Tools in Knowledge-Based Systems*, is divided into four main parts and includes the contributions having a basic and methodological nature, along with those concerning knowledge modeling, formal tools, and generic tasks of analysis in applied AI. There are sections on fuzzy knowledge representation and inference, qualitative reasoning, evolutionary computing, and multiagent systems, among others.

One of the most frequent deficiencies in the majority of methodological developments lies in ignoring the conclusive step about how to render the models operational with the final result of an implemented system. We believe that this fact accounts for a considerable lack of credibility towards AI among researches on the outside, who feel that it has failed in that it has not made enough inroads into real-world applications. Consequently, AI researchers are sometimes seen as *just blowing smoke*. It is still common to find journal articles that do not support claims on rigorous experimental evidence or that only show solutions to toy problems by way of validation.

In the second volume, with the title *Tasks and Methods in Applied Artificial Intelligence*, we have included the contributions dealing with aspects that are more directly relevant to application development. These contributions are grouped into five parts: generic tasks of synthesis and modification, machine learning, applied AI and Knowledge-Based Systems in specific domains, and validation and evaluation criteria.

The editors are also aware of the grand challenges for AI concerning artificial behavior for agents that have to deal with the real world through perception and motor actions. Nowadays, there is an enormous lack of balance between existing AI systems in some aspects of their competence. Whereas in some formal microworlds AI systems have reached the highest human level of competence —the recent success of chess-playing systems being a paradigmatic example—, or there are knowledge-based systems exhibiting human expert competence in narrow technical domains such as medical diagnosis, etc., few systems exist surpassing the competence of a cockroach, for instance, in moving around pursuing a goal in an unstructured world. This enormous distance between pure abstract intellectual tasks at one end, and those that involve sensorimotor interaction with the physical world at the other, calls for an emphasis on research on robotic agents.

Since the current state of affairs is partly due to the Turing vision of a disembodied, abstract, symbol-processing intelligence, new proposals —such as those put forward by Harnad or Brooks— are worth consideration. Robotic capacities including the ability to see, grasp, manipulate, or move have been added to an extended version of the Turing test. The symbol grounding problem has been approached by the physical grounding hypothesis: grounding a system's representations in the physical world via sensory devices with the result of emergent functionalities. Taking the biological paradigm seriously implies building on top of an integrated and distributed sensorimotor system, since the coordination of our movement is done mainly in an unconscious way, relying on perception without central processors coming into play. Neural networks have proven to be an adequate paradigm for approaching this kind of problem as well as others at the subsymbolic level. We believe that the connectionist and symbolic perspectives to AI should be taken as mutually supporting approaches to the same problems, rather than as competitive areas, as is often the case. Hybrid systems integrating both perspectives appear to be the right track to follow.

This emphasis on perception and robotics has obtained a satisfactory response in terms of the number of submitted papers, as compared with previous conferences.

Consequently, a section on perception is included in Volume I, and in Volume II more than 20 papers can be found in sections devoted to perceptual robotics, robot motion planning, and neurofuzzy approaches to robot control.

The papers included in this volume were presented at IEA/AIE-98 which was held in Benicàssim, Castellón, Spain on June 1-4, 1998. The event was sponsored by the *International Society of Applied Intelligence* —which promotes this conference series—, Universidad Jaume I de Castellón —the hosting institution— and Universidad Nacional de Educación a Distancia, in cooperation with several international associations such as AAAI, ACM/SIGART, ECCAI, and AEPIA, among others. Support for this event has been provided by Fundació Caixa Castelló-Bancaixa, Ministerio de Educación y Ciencia, Fundació Universitat Empresa of the Universidad Jaume I, and Silicon Graphics Computer Systems.

We would like to express our sincere gratitude to the members of the organizing and program committees, to the reviewers, and to the organizers of invited sessions for their invaluable effort in helping with the preparation of this event. Thanks also to the invited speakers, Michael Brady and Bob J. Wielinga, with particular gratitude to Roberto Moreno-Díaz, for their original papers given as plenary lectures and appearing in this book. Thanks also to Moonis Ali, president of ISAI and IEA/AIE-98 general chair, for his constant support. The collaboration of the Technical Committee on Robot Motion and Path Planning of the *IEEE Robotics and Automation Society* deserves a special mention, as well as Toshio Fukuda, president of this society, for his help in the review process. Also, thanks to Springer-Verlag and particularly to Alfred Hofmann for an already long and fruitful collaboration with us. We sincerely thank all authors for making the conference and this book possible with their contributions and participation.

Finally, the editors would like to dedicate this book to the memory of Núria Piera, who promoted research on qualitative reasoning across Spain and Europe and could not see by herself the success of her last organized session, since she had to move to her definitive dwelling.

The theme for the 1998 conference was *New Methodologies, Knowledge Modeling and Hybrid Techniques*. Our focus has been on methodological aspects in the development of KBS's, knowledge modeling, and hybrid techniques that integrate the symbolic and connectionist perspectives in AI applications. The global assessment of the contributions contained in these two volumes is reasonably positive. They give a representative sample of the current state of the art in the field of Applied Artificial Intelligence and Knowledge Engineering and they clearly illustrate which problems have already been solved or are on the way to being solved and which still present a challenge in the serious enterprise of making Applied Artificial Intelligence a science and an engineering discipline as unequivocal and robust as physics or matter and energy engineering. We hope that these volumes will contribute to a better understanding of these problems and to expedite the way to their solution for the well-being of humankind with the advent of the third millennium.

Angel Pasqual del Pobil
José Mira Mira

March 1998

Table of Contents, Vol. I

1 Methodological Aspects

2 Knowledge Modeling

3 Formal Tools

Fuzzy Knowledge Representation and Inference

Uncertainty, Causal Reasoning and Bayesian Networks

Qualitative Reasoning

Neural Networks

Evolutionary Computing

Object-Oriented Formulations

Hybridization Techniques

Context-Sensitive Reasoning

Multi-Agent Systems

Divisible Problems

4 Generic Tasks of Analysis

Perception

Natural Language Understanding

System Identification

Monitoring

Fault Diagnosis

Predictive Models

Table of Contents, Vol. II

1 Synthesis Tasks

Spatial, Temporal and Spatio-Temporal Planning and Scheduling

Motion Planning for Robots

System Configuration

2 Modification Tasks

Knowledge-Based Control Systems

Dynamic Systems Supervision

Intelligent Supervision

Predictive Control Systems

Perceptual Robotics

Fuzzy and Neurofuzzy Approaches to Robot Control

Program Reuse

3 Machine Learning

Machine Learning Applications: Tools and Methods

Inductive and Deductive Strategies

Case-Based Reasoning

Learning Advances in Neural Networks

4 Applied Artificial Intelligence and Knowledge-Based Systems in Specific Domains

Multimedia

Human-Computer Interaction

Decision Support Systems

Tutoring Systems

Connectionist and Hybrid AI Approaches to Manufacturing

Modeling and Simulation of Ecological/Environmental Systems

Reports on Real Applications with Significant Findings

5 Validation and Evaluation Criteria

Simulation

Knowledge Technology: Moving into the Next Millennium

B.J. Wielinga and A.Th. Schreiber

Department of Social Science Informatics
University of Amsterdam
Roetersstraat 15, NL-1018 WB Amsterdam
{wielinga, schreiber}@swi.psy.uva.nl

1. Introduction

Two decades of knowledge engineering research have resulted in a wide range of methods and techniques for the acquisition, modelling, representation and use of knowledge (David, Krivine and Simmons, 1993; Schreiber et al., 1994a). The purpose of this enterprise was mainly to support the construction of knowledge-based systems (KBS) that can perform knowledge-intensive tasks such as diagnosis, configuration, assessment, planning and the like. The recent recognition of the essential role that knowledge plays in organisations and in modern society at large, has triggered questions about the wider applicability of knowledge engineering methods and techniques as a possible basis for a technology for the management, use, explication and creation of knowledge. Many companies possess large amounts of knowledge most of which resides in the heads of their employees or in documents and heterogeneous databases that are difficult to access with general queries. Scientific communities accumulate large amounts of knowledge and record that knowledge in scientific papers, reports and books. Finding the answer to a question in an organisation or in the scientific literature often resembles the search for the proverbial needle in the haystack. The World Wide Web contains enormous amounts of information and knowledge -Alta Vista reports over half a million pages that contain the term "knowledge"-, but finding the relevant pieces of knowledge can be a draconian task.

The general goal of this paper is to assess the knowledge assets that we currently have and to explore how we can employ knowledge engineering technology to use the knowledge resources more effectively than we do today. In other words, the quest of the Knowledge Engineering community is to find ways of employing the full power of the knowledge that is ready at hand.

After a brief impression of what the space of knowledge resources that we are talking about contains, we will review the assets of the knowledge engineering discipline and discuss some of the major issues in knowledge reuse in building knowledge-intensive applications. Subsequently we will attempt to formulate a number of problems that hamper the accessibility, use and reuse of large amounts of knowledge in a more general context. We will indicate some directions towards solutions of these problems. Both Knowledge Engineering (KE) and Knowledge Management (KM) are concerned with modelling knowledge, albeit at different levels of detail. We will argue that the knowledge modelling methods, techniques and tools developed for KE can be applied to KM in a variety of ways.

The term "ontology" will play a central role in this paper. To date this term is used in a variety of ways, varying from just a synonym for a knowledge base to the philosophical notion of "theory of being". Initially we use the term "ontology" in a loose sense, meaning something like a "generic knowledge base", later in the paper we will attempt to give a more precise definition of what an ontology is and what the benefits and the potential of ontologies are in the context of the general problem of knowledge sharing and reuse.

2. The Knowledge Resources

The amount of knowledge that mankind has documented over the last three millenniums is enormous and is growing at a non-linear rate. Our concern here is, however, the fact that knowledge in machine-readable form is becoming available to the average computer user at an even faster rate. Current Information and Communication Technologies (ICT) already put Giga bytes of textual and multi-medial information at the users' disposal through a simple command or by inserting a CD-ROM. It is not unreasonable to expect that in the first decade of the next millennium Tera bytes of information can be called up at will. However, information only becomes knowledge when it can be interpreted and can subsequently be used by an agent to answer some question or to achieve some goal. To support that interpretation and use, knowledge technology requires some handles on the structure and meaning of the information.

Hence, we will mainly restrict ourselves in this paper to sources of knowledge that have some internal structure that is computationally exploitable. Images, speech documents and free natural language documents will not be considered here. The knowledge resources that we will mention all have some (semi-) formal structure, albeit that this structure may not be imposed all the way down to the level of the detailed knowledge items.

- **Knowledge bases**: many thousands of KBS have been built, many of these are proprietary or not accessible, some of them are public domain but unlikely to be reusable.
- **Specialised Ontologies:** a number of ontologies and ontology libraries for specific domains are available.
- **General Ontologies:** a number of general ontologies exist covering a broad range of common sense knowledge (Cyc, WordNet, EuroWordNet).
- **Thesauri**: although thesauri are mainly intended for indexing and classification purposes, they also are a rich source of knowledge. Many

large thesauri and other classification systems are available (e.g. AAT, ULAN, TGN, MeSH, ICD96, LCSH).

- **Product Model Application Protocols** (STEP): the STEP community is developing a large set of data models for product modelling (e.g. AP281 for modelling ships).
- **Large Databases:** for specific fields of science and technology large databases exist which not only contain data, but often also capture generic knowledge, albeit in forms that are not always easily accessible. For example GENBANK, a large data base of molecular biology data, also contains a large taxonomy of organisms.
- **Knowledge on the World Wide Web:** The World Wide Web (WWW) contains many other forms of knowledge which are structured in some form or other. Recently various proposals have been made to annotate and structure information on the WWW using extensions of HTML (Benjamins and Fensel, 1998; Heflin et al. 1998) or XML (Bray et al. 1998).

An area that has not received much attention in the knowledge engineering community is knowledge represented in diagrammatic form. In many domains, especially technical and engineering domains, diagrams are an important means of documentation and hence a source of knowledge. Maintenance documentation of aircraft, for example, largely consists of annotated diagrams and schema's. SGML versions of such documents are becoming available and should be considered a new challenging source of knowledge.

3. The Assets of the Knowledge Engineering Discipline

Knowledge engineering (KE) research has evolved in many respects since the early days of simple rule-based expert systems. A major advance has been the introduction of the "knowledge-level" in the mid-eighties. Newell introduced the "Knowledge level" as an implementation-independent level which allows a conceptual description of problem solving behaviour and knowledge structures to sustain that behaviour. The form of the knowledge nor its location are of relevance at the knowledge level. It does not matter whether the knowledge resides in the head of someone, is documented in a book, or is represented in an information system.

A number of knowledge modelling methodologies have been developed based on the knowledge-level notion (Schreiber & Birmingham, 1996). Here, we briefly describe CommonKADS as a typical example of the knowledge modelling approach (Schreiber et al., 1994).

A CommonKADS knowledge model contains a number of knowledge *types*. These knowledge types play different roles in problem solving. The knowledge model in knowledge engineering consists of five major knowledge types: tasks, methods, inference knowledge, domain-knowledge schema's, and domain knowledge:

A *task* specifies a goal in a functional way, indicating input/output of a task and logical dependencies between task I/O. An example task is diagnosis, in which the input is a complaint about system behaviour and the output is a fault category.

A *method* is a prescription for solving a certain (sub-)problem. Example methods are generate & test, propose & revise, etc. Methods come in fact in two varieties. A *generic method* is a method description that is independent of a particular task. A *task method,* on the other hand is a method that is applied to a particular task, One can see it as an instantiation of a generic method in which generic terms have been replaced by task-specific terms.

Inference knowledge describes the basic inferences that we want to make in the domain. An inference operates on some input data and has the capability of producing a new piece of information as its output. Inferences operate over knowledge elements. These knowledge elements are described as *dynamic* knowledge roles (e.g. the input problem, intermediate results, the solution). In carrying out their operation inferences also make use of knowledge elements that are not effected by the operation. These knowledge elements are described as *static* knowledge roles

A *domain-knowledge schema* is a schematic description of the structure of the domain knowledge used to solve a problem. A domain-knowledge schema has a similar function as a data model in traditional software engineering, but it is more complex due to the meta-character of knowledge ("knowledge about knowledge"). The term "ontology" is often used as a synonym for domain-knowledge schema.

Domain knowledge is the collection of concepts, relations and facts that are used by the inferences.

For each of these knowledge types modelling languages and notations, both formal and informal, have been developed (Fensel and van Harmelen, 1994). Within the framework of the CommonKADS methodology a modelling language (CML, Schreiber et al., 1994b) has been developed which allows the semi-formal description of a knowledge model. CML has been extensively used to describe large ontologies for a number of technical domains (Schreiber et al., 1995). A special mapping mechanism in CML allows the construction of layered ontologies in which higher layer elements represent more abstract knowledge types.

Knowledge engineering can be a laborious process. Therefore, much attention has been devoted in KE research to techniques for knowledge reuse. Both tasks, methods as well as domain-knowledge schema's can be reused, although the nature of reuse varies.

For problem-solving tasks, a task typology has been developed based on characteristics of the problem the tasks are set out to solve. Example task types are diagnosis, design, configuration and scheduling. The task types are partially ordered in a type hierarchy, e.g. configuration is a sub-type of design. The actual tasks in the application task (sometimes called the "real-life tasks"), however, rarely correspond to one single task type. Typically, an application task requires a combination of task types. For particular task types, the knowledge engineering community has created libraries of methods, such as the CommonKADS library (Breuker and Van de Velde, 1994) in which a large collection of methods is listed.

With respect to reuse of a domain-knowledge schema, the situation is more complicated. It is an empirical fact that the structure of domain knowledge is partially determined by the way it is used in the problem-solving process ("the relative interaction principle"). This dependency hampers reuse, because it means that schema reuse depends on the task and/or method context for which it was first developed. However, experiments have shown that it is possible to define schema's at different levels of specificity and connect these through partial mappings. This levelling of domain-knowledge schema's can be used to identify different domain-knowledge types with different "reusability status". Currently, we consider six levels of schema specification:

An *application schema* is the most specific domain-knowledge schema. It contains precisely those domain-knowledge types that are required by a certain application.
A *method-specific* schema introduces the representation required by a certain (subset of) methods. For example, a causal-covering method requires a causal-network representation.
A *task-specific* schema contains those conceptualisations that are inherent to a task. For example, in a design task the notions of component and constraint need always to be present.
A *domain-specific* schema describes the conceptualisations in a domain that are independent of the task and/or method employed. For example, we can have a structural description of a device that is both used in design and in diagnosis.
An *inter-domain* schema generalises over domains and provides generic descriptions that can be found in classes of domains. For example, in technical domains within the process industry, reusable domain-knowledge types can be identified.
A generic schema describes a number of definitions that are supposed to be more or less universally true. Generic schema's resemble Aristotelian categories. The main difference is that in KE research no claim is made about the completeness of these schema's. Their purpose is more pragmatic than philosophical: the schema should help in making as much as possible reuse of previous experience in knowledge specification.
Recently some efforts have been made to create libraries of schema definitions. The schema types in these libraries vary. For example, the KACTUS project (Schreiber et al., 1995) focused on the last three types of ontologies. In the Sisyphus project (Schreiber and Birmingham, 1996), the identification of the differences between task-specific and method-specific ontologies has been studied.
In addition to schema reuse, domain-knowledge schema's can also be used to support (semi-automatic) translation (mapping) of domain-knowledge instances of one schema into a instances of another the scheme.

In summary the assets created by two decades of KE research are:

- The knowledge level
- Separation of different knowledge types
- Relative interaction principle and the notion of knowledge roles
- A set of semi-formal and formal knowledge level modelling languages
- Ontologies as classification and description vehicles
- Ontologies as a tower of descriptive meta-levels and as mediator between knowledge sources
- Libraries of Problem Solving Methods.

Knowledge engineering methodologies such as CommonKADS have matured and are increasingly being applied in industrial contexts. It is the more remarkable that so few software tools exist on the market to support these methodologies. Several large EU funded projects have constructed prototype tool kits and workbenches for knowledge acquisition, knowledge analysis and KBS design. Very few of these prototypes have led to commercial tools, PCPACK being one of the few exceptions. It appears to be difficult to design usable tools to support the knowledge engineering process. One of the reasons for this may be that the principles underlying the organisation of large amounts of knowledge are not understood well enough.

4. Multiple Views on Knowledge

One of the central problems in knowledge modelling and information engineering in general has been the representation of multiple viewpoints on the objects that must be represented. Data modeling essentially takes a single-level view of the world. The data model describes in a schematic way the "objects" or "instances" that are of interest. For example, a data model for a database of oil platform components would contain types such as "heat exchanger" together with related attributes and relations/associations. Instances of this type are particular heat exchangers. Although subclass hierarchies can be used to define more abstract properties of a "heat exchanger", data modelling typically allows only one possible interpretation of "heat exchanger".

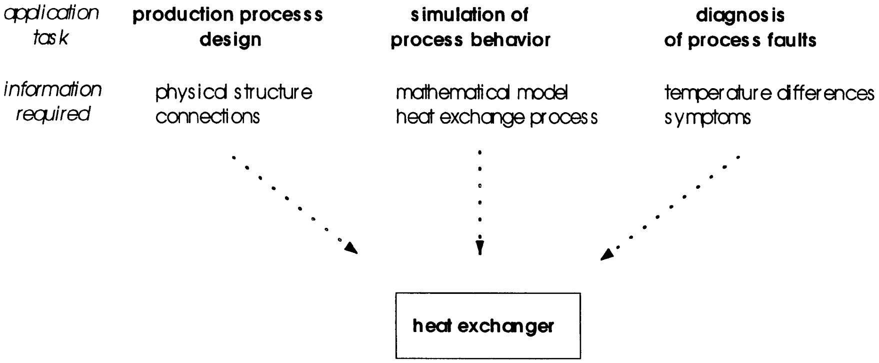


Fig. 1. Different viewpoints on a "heat exchanger"

However, if one wants to reuse a notion like "heat exchanger", it quickly becomes clear that this term can have different meanings in different contexts (see figure 1). For example, from an oil-platform design perspective the physical properties will be the main emphasis: physical dimensions, types of connections, etc. In a diagnostic setting functional properties such as the difference in temperature between the different inputs and outputs are likely to be the prime focus of attention. For dynamic simulation applications, behavioral properties such as mathematical properties of the

heat exchange process would need to be modeled in association with “heat exchanger”.
It is safe to say that reuse of complex, knowledge-intensive data is impossible without taking these different viewpoints into account. This means that we want to be able to look at a data type like ``heat exchanger'' in different ways. This idea is partially present in the ANSI/SPARC architecture of databases. The ANSI/SPARC external views can be used to some extent for the representation of the different viewpoints, such as those on a “heat exchanger”. The main limitation is that the viewpoints themselves do not add new information: all basic information is already present in the conceptual schema. This is not the case in most realistic situations: each viewpoint on an object adds its own context-specific information items, and there is not one universal unified representation for all the viewpoints. Such a unified representation is impossible because the number of viewpoints one can take is in practice unlimited. In other words, objects like ``heat exchanger'' can be used in so many different contexts that it is impossible to develop in advance a complete, unified description of a ``heat exchanger'' that will prove to be sufficient for every possible situation.

This observation about viewpoints on real-world objects has led to the following major principle for domain-knowledge modelling:

> **Principle 1:** *The representation of real-world objects always depends on the context in which the object is used. This context can be seen as a ``viewpoint'' taken on the object. It is usually impossible to enumerate in advance all the possible useful viewpoints on (a class of) objects.*

This principle has important consequences for an approach to reusing knowledge-intensive information items. It means that it is not feasible to strive for one unified representation, that can be used in multiple application settings. Typically, one would expect that application descriptions can be partially reused in another setting, namely exactly those descriptions that share a common viewpoint. For example, in the electrical network domain, the functional model of the network built for a diagnostic application might also be useful for a service recovery application, whereas some other knowledge items are likely to be application-specific for a diagnostic context.
It should have become clear by now that the key to reuse lies in explicating the underlying viewpoints of a representation. This can be seen as the second principle underlying domain-knowledge modelling:

> **Principle 2**: *Reuse of some piece of knowledge requires an explicit description of the viewpoints that are inherently present in this knowledge. Otherwise, there is no way of knowing whether, and why this piece of knowledge is applicable in a new application setting.*

For arriving at such an explicit description of viewpoints, the notion of “ontology” is introduced.

4.1 Ontologies

We use the notion of ontologies as a vehicle for describing the underlying structure of pieces of knowledge. One can see ontologies as a natural next step in the increasing complexity of data models. There is no hard borderline between complex semantically-rich data models and ontologies. Ontologies specify what is called a “conceptualisation”: a way to view the world (Gruber, 1993). Every ontology thus incorporates a particular viewpoint. An ontology contains definitions that provide us with a vocabulary to talk about a domain of discourse. What these definitions look like depends on the language that we have chosen to represent ontological definitions.
Two features are typical of ontologies: the fact that there can be multiple ontologies (= viewpoints) on the same domain, and the fact that we can identify generalisation dimensions of ontologies. We discuss both features in more detail.

Multiple ontologies

Assume we have some artefact such as a ship. One can define multiple viewpoints on a ship. Well-known examples of such viewpoints are the physical structure (“what are the parts of a ship?”) and the functional structure (“how can a ship be decomposed in terms of functional properties?”). Although these two viewpoints often partially overlap, they constitute two distinct ways of “looking” at a ship. The purpose of an ontology is to make those viewpoints explicit. For a design application, such as a CAD application, one would typically need a combined physical/functional viewpoint: a combination of two ontologies. For a simulation application (e.g. modelling the behaviour of a ship) one would need an additional behavioural viewpoint. Many other viewpoints exist such as the process type in the artefact (heat, flow, energy, ...). Each ontology introduces a number of specific ``conceptualisations'', that allow an application developer to describe, for example, a heat exchange process.

Levels of ontologies

Ontologies are not just a flat set of descriptions of conceptualisations. Typically, one can identify several levels of abstractions on which ontologies can be defined. For example, in an electrical network domain we could have a set of ontologies of an electrical network in the Basque country in Spain, describing how the various sub-stations and their sub-parts together make up a network. We can first try to generalise from this ``Basque'' ontology, and try to come up with ontologies for electrical networks in general. A second generalisation step would be to define for those ontologies describing electrical processes in the network a general ontology for electricity processes, independent of whether it is a electrical network or some other artefact in which the process takes place. Yet another level up, we can think about general categories of knowledge structures. What are the general characteristics of things that we can structurally divide up in parts? Or of connections between components?

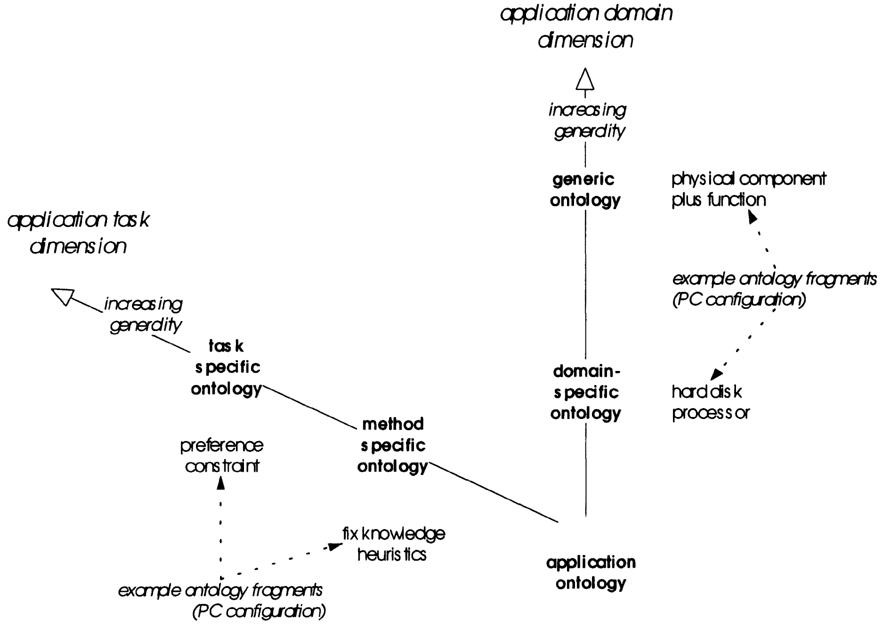


Fig. 2. Generalisation dimensions of ontologies

Figure 2 shows this generalisation dimension in the vertical direction. A second dimension concerns the degree in which an ontology is specific for an application, a problem solving method, a task or is independent of the way in which will be used.

4.2 Reuse and Interoperability of Knowledge Bases

One feature of the ontology approach is that it provides a principled way for supporting interoperability of applications through exchange of knowledge or complex (semantically rich) data. This type of interoperability is shown schematically in figure 3. If one can assume that two applications share the same ontology (which is typically a sub-part of the overall application ontology of an application), then the two applications can exchange information that is based on this shared ontology. The exchange procedures can be very simple or elaborate, depending on the differences in implementation-specific representation choices, but the application developer can be sure that information exchange is always technically feasible. A number of examples of exchange based on shared ontologies exist. For example, in the Sisyphus-VT experiment (Schreiber and Birmingham, 96a) several contributors were able to reuse an existing knowledge base of elevator components and constraints in their application, even if their application was written in an entirely different implementation language. Typically, each application added its own application-specific conceptualisations, based on the different ways in which the design task was realised. Ontologies served the role of explicating which parts of an application knowledge base can be shared, and which parts cannot.

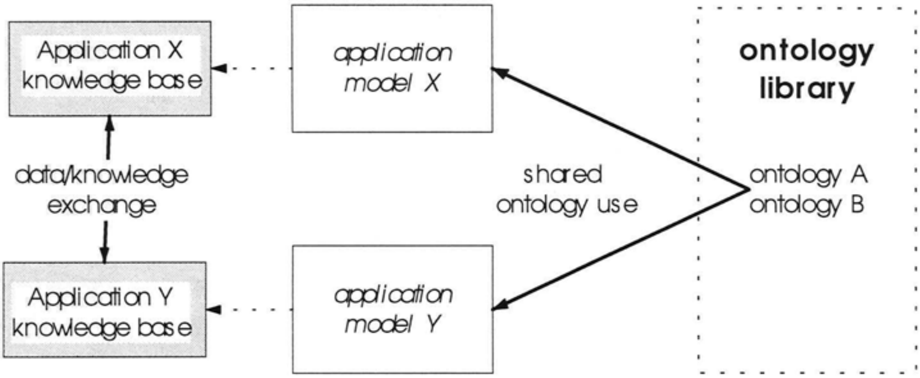


Fig. 3. Sharing an ontology and exchanging data

In a ship-design domain this "shared ontology" approach was used to enable the exchange of ship-design data between the ship designers and the ship assessors, although designers and assessors have quite different ways of looking at a ship design. In another application a shared ontology of an electrical network was employed to reuse a network knowledge-base in a new application. In this latter application the reuse enabled the developers to construct a fully functioning prototype system within one week (see Laresgoiti, 96 for details).

5. Ontologies in a Wider Perspective

Although ontologies and problem solving methods provide a good basis for reuse of knowledge over individual KBS applications, a more general form of reuse and sharing of existing knowledge remains remarkably difficult. Even if the relevant knowledge to solve a problem or to answer a question can be found, it is more often than not in a form that is unsuited for direct use. There are many different reasons for this.

- The knowledge is represented in the wrong language or vocabulary.
- The knowledge is represented from another viewpoint than the one required.
- The knowledge is represented at a different level of generality than is required.
- The knowledge provider has made a number of implicit assumptions about the use of the knowledge.
- Multiple heterogeneous sources of knowledge need to be combined and integrated in order to solve a problem.

Ontologies and comparable representational vehicles have been proposed and are being employed as a potential solution to these and similar problems. In the knowledge engineering community the notion of ontology is usually viewed as a meta-level knowledge base that captures some generic aspects of the knowledge in the domain. In other sub-disciplines of Information Science- to be understood as the general science of the nature of information and its use- other terms are used for similar notions. The STEP community is developing standardised *product models* ("Application Protocols") to capture generic knowledge about a domain such as ships. The database world uses the term *meta-data,* thus emphasising the meta-level character of ontologies. The library sciences heavily rely on the notion of thesaurus as a vehicle to classify the concepts in an area of interest. In other fields of science, technology and the humanities we see the use of taxonomies, standardised glossaries and controlled lists of terms as a means to control the language used for classification and indexing. Last but not least, Philosophy has been concerned with ontology as the theory of distinctions among the entities in the world: the categories of being. All of these approaches have a number of dimensions in common, but differ widely in the approach along those dimensions.

Representation language: Natural language is clearly not precise enough to represent ontologies. The Knowledge Engineering community uses a variety of formal and semi-formal representational languages such as Ontolingua (Gruber, 1993), CML (Schreiber et al., 1994; Schreiber et al. 1998). The STEP community uses a data modelling language EXPRESS. The data base world is working towards UML as a standard language for semantic data modelling. The library sciences use standardised record structures for representing thesauri. The use of these diverse representational languages clearly makes knowledge reuse and knowledge sharing difficult. One possible approach towards solving this problem is to standardise on one single language. The KACTUS project (1996) has considered this option and rejected it. Each language has its own merits and application potential. Ontolingua is good for formal modelling, CML is suitable for semi-formal representation, EXPRESS has some disadvantages from a knowledge modelling point of view, but interfaces well with other engineering applications. The solution pursued in KACTUS was to provide partial translations between languages. For example, the STEP model of a ship (AP218) used in a CAD environment, was translated into a CML based representation used in an assessment application. In a similar vein, the GRASP (1996) project uses a translation of the Art and Architecture Thesaurus (Getty, 1996) into structures which are amenable to inference, as the core of a knowledge base for indexing stolen art objects.

Representational ontology and constraints: In addition to the representation language, the representational constructs from which an ontology is built have to be defined. The Ontolingua approach provides a *frame ontology* that defines a number of well understood knowledge representation primitives. UML and CML, which can be considered an extension of UML, provide a rich vocabulary for representing concepts (objects) and various types of relations. EXPRESS provides a number of constructs that are somewhat more primitive than those in modern knowledge engineering languages. The ISO standard for the representation of thesauri only allows concept definitions in terms of simple attributes and a strict hierarchical sub-super type relation. Linguistically oriented constructs such as synonym and antinym relations, are used in (Euro) WordNet. As was argued in the case of representation language, standardisation on one representational ontology is unfeasible and probably even undesired since our representational vocabulary will change and continue to become richer over the years. What is important however, is that a set of standard *mappings* between the various constructs becomes available. Examples of such mappings have been constructed in a number of cases, but general principles for mappings between representational constructs are still very much a subject of research.

Object versus meta-level: The original notion of ontology was intended to be used for theories at the meta-level: theories that describe what the distinctions and structures are that are used in constructing a knowledge base at the object level. This meta-level character has suffered some inflation since the term ontology is often used for the actual knowledge base of a certain domain. This ambiguity can be understood when we consider knowledge not at just two levels (object and meta) but at multiple levels that all have an object-meta relation between them. Consider for example a taxonomy of organisms as it is used in GENBANK. This taxonomy is a hierarchy of types of organisms and as such it describes the classification structure of the genomic data stored in the database. As such it could be considered to be an ontology. However, if we would construct an ontology that describes how biological taxonomies are constructed in terms of concepts like "kingdom", "species" etc, the taxonomy would be the object level knowledge base and the ontology of taxonomic layers would be the meta-level (see Figure 4). In a similar way, thesauri used for indexing books or art objects, can be viewed as meta-level descriptions of the objects that are classified with the thesauri. From a knowledge engineering perspective, such thesauri are just knowledge bases about a certain domain. What is an ontology at one level, can be viewed as a knowledge base at another level. Current practice in ontology construction does not always make a clear distinction along the object-meta level dimension. Clarifying this dimension and making the object-mata relation explicit would enhance the interoperability and reusability of ontologies and knowledge bases.

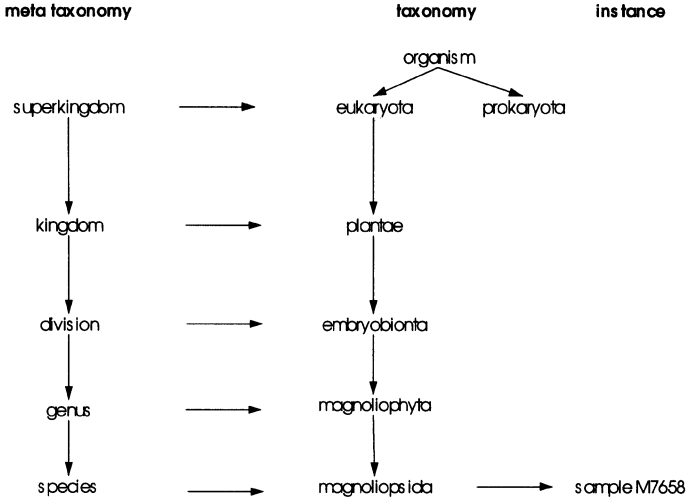


Fig. 4. Ontologies at meta and object level

Hierarchical structure of concepts: Most ontologies and similar vehicles use a hierarchy or taxonomy of concepts as the backbone of their structure. However, the way in which such a hierarchy should be organised is a matter of much debate. Fridman-Noy and Hafner (1997) discuss the top-level hierarchies of ten different ontologies. There appears to be little or no correspondence between these top-level structures. However, some of the distinctive categories -such as abstract versus concrete, or tangible versus intangible- occur in many of the ontologies albeit in different levels in the hierarchy. This lack of consensus on a top-level ontology poses a serious problem for knowledge reuse and interoperability. The cause of the disagreement lies in our opinion in the different viewpoints that the various ontology constructors adhere to and in the fact that these viewpoints remain implicit. As we have argued above, fixing one or more viewpoints on a domain is virtually impossible, hence we should not strive for one fixed hierarchical structure that we all agree upon.

A possible solution direction is to make the viewpoints that underlie a hierarchy of domain concepts explicit and focus on the construction process rather than on the result itself. In the KACTUS project this line of thought was explored by distinguishing different types of ontologies: primary ontologies and secondary ontologies. Primary ontologies represent knowledge about concrete or abstract entities in the world according to a particular set of viewpoints. Secondary ontologies represent knowledge about the viewpoints and distinctions we can impose on the world. As an example consider the viewpoints physical, functional and behavioural that we discussed before in the context of viewpoints on devices such as a heat exchanger. Applying these viewpoints to the notion of "entity" in a primary ontology would result in a top-level hierarchy as shown in the left branch of figure 5.

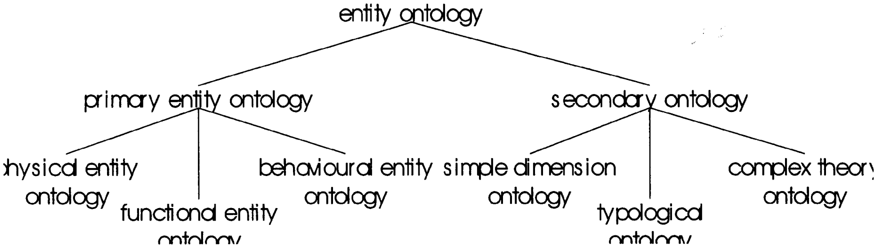


Fig. 5. Applying viewpoints to "entity"

Figure 6 shows a hierarchy of devices used in the process industry generated through the application of a number of viewpoints: functional, process, domain of operation (thermal energy, fluids, electricity). The advantage of this explicit application of viewpoints is that we can always trace back the process of the hierarchy construction and we can map concepts from different hierarchies onto each other. A similar technique is sometimes used in thesauri through the incorporation of "guide nodes". The AAT (Getty, 1996) for example often inserts intermediate nodes which describe what differentiating property was used to generate the sub tree (see Figure 3). Another interesting variant of the principle of explicating the differentiating dimensions is used in the top-level ontology of Euro WordNET (Rodriquez et al., 1998). Here a selected set of concepts is characterised on a fixed set of semantic properties such as origin, function, form. This allows to determine the general semantic class that a concepts in a certain part of the hierarchy belong to.

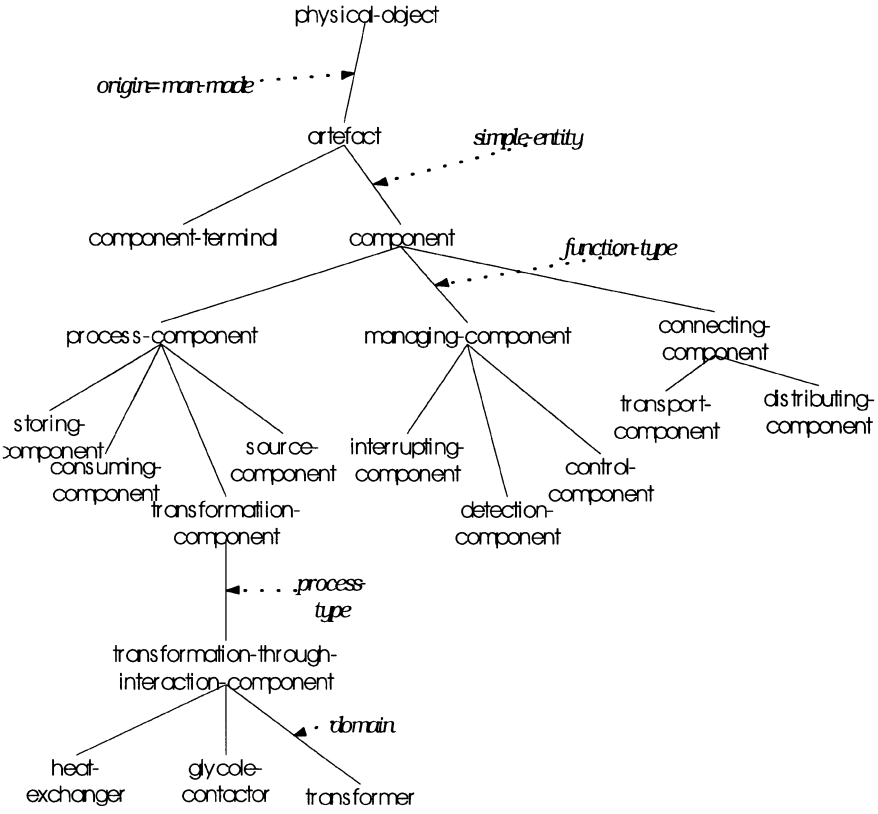


Fig. 6. A taxonomy of devices generated by various viewpoints

The general principle that underlies these approaches is to make the semantic distinctions that were used in construction of a hierarchy explicit part of the ontology, thus enabling a better analysis of what the context of a piece of knowledge is.

6. Knowledge Management

With large amounts of knowledge becoming available in more or less explicit form, the management of this knowledge comes into focus. The current trend to view Knowledge Management largely in an organisational context does not do justice to the fact that the shear size of the knowledge that mankind as a whole possesses needs some form of management itself. It is an important challenge for the research communities concerned with knowledge and its description, to contribute to the knowledge management area. We define Knowledge Management as the collection of those processes that describe and administrate the knowledge assets of an organisation or community and that guide the conservation and enlargement of those assets. Like Knowledge Engineering, Knowledge Management is thus concerned with modelling of knowledge and knowledge-intensive processes. However the practical means for effective Knowledge Management are scarce, if available at all.

The processes that constitute the Knowledge Management activity itself can be viewed as knowledge-intensive problem solving tasks. Knowledge is not only the object of Knowledge Management, but Knowledge Management itself requires knowledge of ways to describe knowledge, to develop knowledge and to maintain knowledge. Again this meta-object nature causes problems in understanding what exactly is being managed. As long as this difficulty exists it is hard to develop methods and techniques for Knowledge Management. In addition, methods for operating on knowledge (explicating, developing, combining, distributing, validating, consolidating) knowledge are as yet not very well understood. We will argue that the Knowledge Engineering discipline can provide some ingredients from which Knowledge Management methods and tools can be built, if they are combined with results of efforts in Information Sciences.

A first, and most important lesson that can be learned from KE is that the Knowledge Level is the right level to describe knowledge. Although aspects such as accessibility, learnability and usability of knowledge are important properties of knowledge in a KM context that may depend to some extent on symbol level properties, the most important aspect of knowledge in KM is its content and not its representation or implementation.

The grain size used for describing knowledge in KM can vary according to the scope and goals of the management activities, but will no doubt be larger than the grain size used for KE. In KM one typically uses notions of collections of concepts, facts, rules, procedures etc. rather than the individual knowledge elements. The classic KE distinction between knowledge about a task and domain knowledge used to perform that task is also a useful one for KM. In addition to the description of the content of knowledge, it is important to know where the knowledge resides, what its availability is and how it will involve in time (Gardner, 1995). Although designed for modelling knowledge at a low grain size, modelling languages like CML (Schreiber et al. 1994) are well suited for the representation of knowledge for KM.
Problem Solving Methods have their pendant in KM as methods for developing, combining, distributing and consolidating knowledge. It would be interesting to investigate whether a library of such models can be constructed in a similar vein as the KE community has constructed libraries of problem solving methods.

Ontologies can play a role in KM at different levels. At the lowest level -the object level- they play an important role as schemata for representation and indexing of information. For example, an ontology of materials and techniques for painting can be used as an index of a library of scientific papers. Thus, ontologies at the object level support the accessibility of knowledge. In many domains such ontologies or thesauri exist or are being constructed, often by researchers with an Information

Science background. The techniques and tools developed for KE can provide support for this construction process.

At a second level an ontology can help the knowledge manager to model the business processes in an organisation. One can envisage libraries of models of processes, functions and roles in various types of organisations. Similar to reference models used for the construction of classical information systems, such a library can greatly enhance the effectiveness of the KM process.

At a third level, we envisage an ontology of knowledge management concepts and tasks. The current literature on KM uses a rather heterogeneous vocabulary. This is not surprising, since the study of knowledge at a higher level of aggregation than sentences or facts, has been largely informal. Whereas KE can build upon the formal terminology and theories of logic and Artificial Intelligence, KM has to resort to informal notions in natural language with terminological ambiguity as a result. As an example consider the terminology for knowledge collections at different level of detail - the "knowledge span" dimension as Wiig (1993) calls it.

Knowledge span (Wiig)	KE terminology	Example
(no term)	knowledge field	Medicine
knowledge domain	knowledge region	Internal Medicine
knowledge region	knowledge section	Endocrinology
knowledge section	knowledge domain	Growing Disorders
(no term)	knowledge structure	Taxonomy of diseases
knowledge segment	task knowledge	Diagnosis of growing diseases

Table 1

Table 1 shows the terminology for knowledge areas as used by Wiig compared to the terminology used in KE (Abu-Hanna, 1993). We are not arguing in favour of one or the other set of terms, but we want to illustrate the lack of agreed upon terminology. An ontology in which categories and concepts of knowledge at the aggregation level that is required for KM, could alleviate this problem. Such an ontology would be the starting point for theories and methodologies for KM.

7. Conclusions

Various fields of information science dealing with knowledge face the fundamental problem of imposing structure upon the enormous collection of knowledge items that is already available. Twenty years of knowledge engineering practice has taught us that knowledge is only useful if it is known what its context and scope is and what viewpoint it represents. We have argued in this paper that it is of crucial importance to characterise knowledge explicitly along a number of dimensions:

- Level of generality: is the knowledge at the top-level, generic, domain specific or application specific?.
- Task specificity: to what extend is the knowledge specific for a class of tasks or problem solving method, and if so what tasks and methods are supported?.
- Scope: what type of knowledge area does the knowledge belong (a field, a domain, a region)?
- Coverage: how well does the knowledge cover the area?
- Content area: to what content area does the knowledge belong (e.g. medical, oil production, art and antiques)?
- Viewpoint: what viewpoints are represented in the knowledge structures (e.g. technical, geometrical).
- Assumptions: what assumptions underlie the knowledge (e.g. time independence).
- Links: what links to other knowledge are there (e.g. a meta-object relation, multiple viewpoints relation)?
- Rigour: to what extend is the knowledge include a "full" axiomatization
- Representational commitments: what are the basic representational constructs from which an ontology or knowledge base is built.

Without such characterisations reuse, knowledge sharing and interoperability is difficult to achieve. A crucial problem is whether a unified ontology for these dimensions can be constructed and whether a consensus can be achieved about such an ontology. Once such a consensus is achieved we can start to develop mappings between knowledge bases and ontologies with different characterisation and ultimately work towards integration of knowledge from various sources. An important problem related to that of characterising knowledge, concerns the principles for organising knowledge in structures of a higher grain size than concepts, relations and axioms. Terms like module, theory, model, taxonomy are used in a variety of ways. Describing and reasoning about knowledge at a higher level of aggregation requires a new formal vocabulary and new theoretical frameworks. A systematic development of such terminology and its formalisation should be high on the research agenda.

These problems bring the knowledge engineering and the knowledge management community together. Knowledge sharing an reuse can only be achieved when knowledge repositories are annotated with and structured according to a consistent set of knowledge characterisations. Knowledge management is concerned with the mapping of these characterisations within an organisational context and with the measures that influence that mapping. Knowledge engineering should be concerned with the ways in which knowledge items can be described by such characterisations, what types of knowledge can be used in what context, what constraints a particular characterisation entails and how knowledge with different characteristics can be mapped and integrated. Other branches of Information Science are concerned with similar issues and are developing their own solutions. The time has come to join forces and to attempt to establish a common framework to organise our knowledge in the digital age to come.

Acknowledgements

Jacobijn Sandberg has contributed to this paper through lively discussions and invaluable comments on earlier versions. Many people have contributed to the ideas developed in this paper. The participants in the KADS-I, KADS-II and KACTUS projects are gratefully acknowledged. This work was partially supported by the European Commission through the KACTUS, GRASP, and TRACKS projects.

References

A. Abu-Hanna, Multiple Domain Models in Diagnostic Reasoning. Ph.D. thesis, University of Amsterdam, 1993.

V.R. Benjamins and D. Fensel (1998). Community is Knowledge! In: (KA)2. Proceedings of the 8th Workshop on Knowledge Engineering Methods and Languages (KEML'98). University of Karlsruhe, 1998.

T. Bray, J. Paoli, C.M. Sperberg-McQueen (1998). Extensible Markup Language (XML). W3C (1998), http://www.w3.org/TR/1998/REC-xml-19980210.html.

J.M. David, J.P. Krivine and R. Simmons (Eds.), Second Generation Expert Systems. Berlin: Springer Verlag, 1993.

D. Fensel and F. van Harmelen, A comparison of languages which operationalise and formalise KADS models of expertise. The knowledge Engineering Review, 9, (1994), 105-146.

Getty Art History Information Program, Art and Architecture Thesaurus, Second Edition. Santa Monica, 1996.

GRASP (1996). The GRASP project, http://www.arttic.com/GRASP/.

J. Heflin, J. Hendler, S.Luke (1998). Reading Between the Lines: Using SHOE to discover Implicit Knowledge from the Web. Submitted to AAI98; http://www.cs.umd.edu/projects/plus/SHOE.

T. Gruber, (1993). A translation approach to portable ontology specifications. Knowledge Acquisition, 5:199-220,1993.

KACTUS (1996). The KACTUS Booklet, University of Amsterdam, 1996.

I. Larisgoiti, A. Anjewierden, A. Bernaras, J. Corera, A. Th. Schreiber, B.J. Wielinga. Ontologies as Vehicles for Reuse: a mini-experiment. In: B.R. Gaines, M. Musen (Eds.), Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems workshop. Calgery: SRDG Publications, 1996.

H. Rodriguez, S. Ciment, P. Vossen, L. Bloksma, A. Roventi, F. Bertagna, A. Alonge (1998). The Top-Down Strategy for Building Euro WordNet: Vocabulary Coverage, Base Concepts and Top Ontology. Computers and Humanities, to appear.

A. Th. Schreiber, B. J. Wielinga, J. M. Akkermans, W. Van de Velde, and R. de Hoog (1994a). CommonKADS: A comprehensive methodology for KBS development. *IEEE Expert*, **9**(6):28–37, December 1994.

A. Th. Schreiber, B. J. Wielinga, J. M. Akkermans, W. Van de Velde, and A. Anjewierden (1994b). The CommonKADS conceptual modelling language. In L. Steels, A. Th. Schreiber and W. Van de Velde (Eds.) A future for Knowledge Acquisition. Berlin: Springer Verlag, 1994.

A. Th. Schreiber, B.J. Wielinga, W.N.H. Jansweijer (1995). The KACTUS View on the "O" word. In: D. Skuce, N. Guarino and L. Bouchard (Eds.), Proceedings of the IJCAI Workshop on Basic Ontological Issues in Knowledge Sharing (pp. 15/1-15/10). Canada:University of Ottawa, 1995.

A. Th. Schreiber and W. P. Birmingham. The Sisyphus-VT initiative. Int. J. Human-Computer Studies, 44(3/4):275–280, March/April 1996. Editorial special issue.

R. van der Spek and R. De Hoog. A framework for a Knowledge Management Methodology. In: K. M. Wiig (Ed.), Knowledge Management Methods: Practical Approaches to manage knowledge. Arlington, TX: Schema Press, 1995.

K. M. Wiig, Knowledge Management Foundations. Arlington, TX: Schema Press, 1993.

In Search of a Common Structure Underlying a Representative Set of Generic Tasks and Methods: The Hierarchical Classification and Therapy Planning Cases Study

J.C. Herrero and J. Mira[1]

Dpto. de Inteligencia Artificial, Facultad de Ciencias. UNED
c/ Senda del Rey s/n. 28040 - MADRID, SPAIN
e-mail: jmira@dia.uned.es

Abstract. CommonKADS methodology for the analysis of human expertise and the development of knowledge based systems yields as a result a library of generic tasks and problem solving methods (PSM's), to be used as components of a conceptual model; later on the corresponding formal and design models have to be made. Currently, given the conceptual model, a complete description that explains how we can obtain the program code is not available yet, though important efforts have been made in order to describe such transitions. Besides, an immediate question is how much these descriptions depend on the generic task.

Our conjeture in this article is that in addition to the libraries of generic tasks and PSM's, there is an underlying structure indeed, a common structure for most of these tasks, at least for the diagnosis and planning tasks. In these cases, we can describe a constructive method for obtaining the operational code for the task and the associated PSM. That is, if we describe the generic task in a suitable way, in terms of natural language, we can establish relationships between this description and some structure we can represent by means of hierarchic graphs. Since we can also establish relationships between the program code and such graphs and both relationships are constant and reversible, we can always use this method to obtain the program code, given the generic task and inversely, to obtain the graph and the model at the knowledge level from the code.

1 Introduction

In his influential paper on the knowledge level, Allen Newell said [9]: "Each [computational] level is defined in two ways. First, it can be defined autonomously... Second, each level can be reduced to the level below... Each aspect of a level... can be defined in terms of systems at a level below... Some intrincate relations exist between and within levels, any instantiation of a level can be used to create any instantiation of

[1] Author to whom all the correspondence must be sent

the next higher level". This describes very well the situation, when we try to compute: we have the knowledge on our problem and we have to reduce it to the program code. Later on, we have to retrieve the knowledge when we obtain the computational results, and we have to beware of making mistakes when we instantiate the upper level by using the lower level instance [6].

The question is: how do we find the reduction road in every case? And the immediate sub-question is: do we have to find it again in every new case? Some trends broach the problem by modelling the knowledge level by means of generic task structures, based on a generic tasks library: any problem we try to compute could be described in terms of more simple problems to solve, in terms of a structure of generic tasks and associated problem-solving-methods (PSM's). There have been two main investigations on generic tasks: Chandrasekaran's works [3] and KADS [15, 12] and CommonKADS [11] methodologies.

According to KADS, as a result of the analysis of the problem that we have to compute, we obtain the expertise model, that consists of two main models: the domain knowledge model and the generic task model. However, the problem we aim remains: there is not any explicit procedure that make us able to build the program given these knowledge level models. The same can be said about CommonKADS and the more recent efforts concerning the development of libraries of generic tasks and reusable problem-solving-methods with explicit ontologies [2]: the transitions between conceptual, and formal models and the final program code are still an open problem.

In this paper we are going to see that it is possible to describe a procedure to reduce the knowledge level to the level below, that is, to describe a procedure to obtain program code from the knowledge level models of a generic task and a method, with the corresponding assumptions concerning the model of the domain knowledge and the knowledge acquisition strategies associated to the PSM. We shall need a suitable natural language description for the knowledge level model and we shall lean on some specific structure, represented by hierarchic graphs. Then we must find relationships between this structure and the knowledge level description, and on the other side, between the structure and the symbol level elements, i.e. the programming code.

The rest of this article is structured as follows. In section 2 we present a description of the hierarchic graphs. Then, in section 3 we study the case of protocols as design plans. First, the task is described in natural language. After this, the correspondence with a hierarchical graph is establised and finally the graph is interpreted en terms of a programming language. Thus, the reduction process is completed.

In section 4 we repeat the method for an analysis task (the hierarchical classification in the Chandrasekaran's sense). We start again giving precision and content to the entities of the domain of interest and the selected PSM (establish and refine), explaining their meaning at the knowledge level and then proposing a computational description, via the graphs of section 2, which again provide an unequivocal link to the symbol level.

The article ends with a reflection on the long term conjecture underlying our work. Namely, that similar descriptions can be found for many other generic tasks and PSM's, because the graph structure of section 2 is invariant, provided some restrictions in the knowledge elicitation process.

2 Hierarchic Graphs

Figure 1 is an instance of a hierarchic graph. A graph [5, 10, 16] is hierarchic if it is connected, directed, finite, perhaps with some cycles, and there is always a node (the starting node) whose descent is the rest of the nodes (and maybe itself). We use some symbols to represent the elements of the graph: the graph itself by $\mathbf{P}_k$, the nodes by $\mathbf{S}_i$, the arcs by $\mathbf{A}_i$ and something associated with every node by $\mathbf{M}_i$.

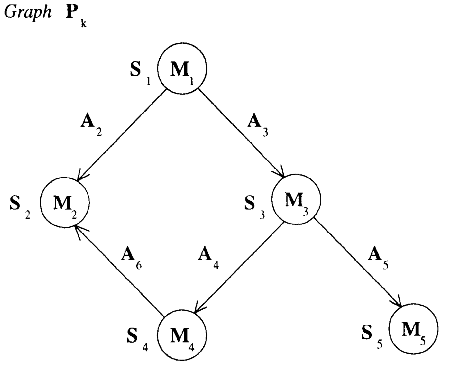


Fig 1. Hierarchic graph instance

We consider these symbols are sets whose elements describe the graph. So we can say there is a set **D** of descriptions, such as

$$\mathbf{D} = \mathbf{P} \cup \mathbf{A} \cup \mathbf{S} \cup \mathbf{M}$$

that is, there are sets like $\mathbf{D}_k \in \wp\mathbf{D}$), and *we* consider (i.e. *we* interpret) their elements $d_{ki} \in \mathbf{D}_k$ are propositions that describe the graph. But, in order to describe the graph completely, we relate a node $\mathbf{S}_i$ to its successors by means of Φ recursively:

$$\Phi (S_i \cup P_k , \mathbf{n}) = S_s \cup P_k$$

where *n* is an integer that expresses some order in the successors, and $\mathbf{S}_s$ is a successor of $\mathbf{S}_i$. We also relate a node $\mathbf{S}_i$ and an arc that start at the node $\mathbf{S}_i$ and points to the successor $\mathbf{S}_s$, by defining Γ:

$$\Gamma (S_i \cup P_k , \mathbf{n}) = S_i \cup P_k \cup A_s$$

and we also relate these to the succesor $\mathbf{S}_s$ of the node $\mathbf{S}_i$ by defining φ :

$$\varphi (S_i \cup P_k \cup A_s) = S_s \cup P_k .$$

Therefore

$$\Phi (S_i \cup P_k , n) = \varphi(\Gamma (S_i \cup P_k , n)).$$

Finally, for every node S_i , we define μ, like this:

$$\mu (S_i \cup P_k) = M_i \cup P_k .$$

It is also easy to see we can obtain a more complex graph where there is more than one starting node, simply by joining several hierarchic graphs [7]. If we join two given hierarchic graphs, $\mathbf{P}_k$ and $\mathbf{P}_q$, some nodes belonging to $\mathbf{P}_k$ have now successors belonging to $\mathbf{P}_q$ or vice versa; for instance if $\mathbf{S}_i$ belongs to graph $\mathbf{P}_k$ and $\mathbf{S}_r$ belongs to graph $\mathbf{P}_q$, and by joining these two graphs $\mathbf{S}_r$ becomes a successor of $\mathbf{S}_i$, then

$$\Phi (S_i \cup P_k , n) = S_r \cup P_q$$

$$\Gamma (S_i \cup P_k , n) = S_i \cup P_k \cup A_r$$

$$\varphi (S_i \cup P_k \cup A_r) = S_r \cup P_q .$$

The same applies for every junction in the more complex graph.

In terms of *generic task, nodes represent domain knowledge*, and arcs permitted inferences. Domain knowledge and inferences knowledge are components of expertise model as we can find them traditionally described [12] as well as in more recent issues [14], as components of conceptual model. Besides, we need a third component: computational flow, procedural knowledge or the generic task itself.

First, in order to obtain a more simple expression for that third component, it is easy to see that we can rewrite the previous expressions in the following terms

$$\mathbf{h(Mi \cup Pk , n) = Ms \cup Pk}$$

$$\mathbf{g(Mi \cup Pk , n) = Mi \cup Pk \cup As}$$

$$f(M_i \cup P_k \cup A_s) = M_s \cup P_k$$

so that

$$h(M_i \cup P_k , n) = f(g(M_i \cup P_k , n)).$$

These expressions are analogous to the previous ones, but we refer to **M** instead of **S**.

Now, computation has some input, we can represent by sets $\mathbf{A'}_i \in \wp (\mathbf{A'})$, and we enlarge the domain knowledge representation by $\mathbf{D'} = \mathbf{D} \cup \mathbf{A'}$, besides we need to define some relation between inputs and inferences, by means of κ:

$$\kappa (A'_i) = A_r .$$

As to our model, it is true that

$$\exists_1 A_s \in \wp (A) :$$

$$[g(M_i \cup P_k , n) = M_i \cup P_k \cup A_s \wedge f(g(M_i \cup P_k , n)) = M_s \cup P_k]$$

and since $\mathbf{A}_s$ is unique, we say that each $m_s \in \mathbf{M}_s$ is true if and only if

$$\exists A'_{\mathbf{r}} \in \wp\,(A') : [\, \kappa\,(A'_{\mathbf{r}}) = A_s \,].$$

Let us define ∇, recursive, that verifies both expressions, i.e.,

$$\nabla\,(A'_{\mathbf{r}} , M_{\mathbf{i}}) = M_s \Leftrightarrow [\, \exists_1\, A_s \in \wp\,(A), \exists \mathbf{n} \in N :$$

$$[\, \kappa\,(\mathbf{A'}_{\mathrm{r}}) = \mathbf{A}_{\mathrm{s}} \,] \wedge$$

$$[\, g(M_{\mathbf{i}} \cup P_{\mathbf{k}} , \mathbf{n}) = M_{\mathbf{i}} \cup P_{\mathbf{k}} \cup A_s \wedge f(g(M_{\mathbf{i}} \cup P_{\mathbf{k}} , \mathbf{n})) = M_s \cup P_{\mathbf{k}} \,]]$$

or else

$$\nabla\,(\mathbf{A'r} , \mathbf{Mi}) = \emptyset.$$

For the starting node, it is true that

$$\exists A'_{\mathbf{q}} \in \wp\,(A') : [\nabla\,(A'_{\mathbf{q}} , \emptyset) = M_1 \,]$$

that is, inputs that set the system to their initial state; then ∇ applies recursively.

Depending on the choice of κ several kinds of computation may result. A simple case can be obtained, for instance, if we disregard probabilistic systems, fuzzy, and so on; we could choose

$$\kappa\,(A'_{\mathbf{i}}) = A_{\mathbf{v}} \Leftrightarrow [\, [\, \forall \mathbf{a'}_{\mathbf{ij}} \in A'_{\mathbf{i}} , \exists\, \mathbf{a}_{\mathbf{vr}} \in A_{\mathbf{v}} : \mathbf{a'}_{\mathbf{ij}} \Rightarrow \mathbf{a}_{\mathbf{vr}} \,] \wedge$$

$$\forall \mathbf{a}_{\mathbf{vr}} \in A_{\mathbf{v}} , \exists\, \mathbf{a'}_{\mathbf{ij}} \in A'_{\mathbf{i}} : \mathbf{a'}_{\mathbf{ij}} \Rightarrow \mathbf{a}_{\mathbf{vr}} \,]\,]$$

that immediately lead us to the following expression, if the elements belonging to sets are supposed to be propositions:

$$[f(M_{\mathbf{i}} \cup P_{\mathbf{k}} \cup A_s) = M_s \cup P_{\mathbf{k}}] \Leftrightarrow$$

$$[\, \mathbf{m}_{\mathbf{i1}} \wedge \mathbf{m}_{\mathbf{i2}} \wedge \ldots \wedge \mathbf{m}_{\mathbf{iu}} \wedge \mathbf{a}_{\mathbf{s1}} \wedge \mathbf{a}_{\mathbf{s2}} \wedge \ldots \wedge \mathbf{a}_{\mathbf{sn}} \Rightarrow \mathbf{m}_{\mathbf{s1}} \wedge \mathbf{m}_{\mathbf{s2}} \wedge \ldots \wedge \mathbf{m}_{\mathbf{sv}}].$$

That yields a deducible system, using the first order logic.

3 Protocols as Design Plans and the "Act-Check-Decide" Method

We are going to broach the protocol problem in medicine [1], from the viewpoint of the generic tasks modelling, and we shall propose and prove a method to obtain the program from a suitable natural language description given this specific model and problem.

As a result of medical research and experience on a specific disease, there are protocols that describe the available strategies for taking care of the patients. After a disease is diagnosed, there may be one or several protocols available as well as choices for every one of them. When the protocol is chosen, we know the routine steps that should lead the patient to retrieve the health, as much and soon as possible. In the best case, we shall give up the protocol because of this. However, depending on the patient response, there could be some reasons to either change the protocol and enter another one for the same disease or to give it up because of the lack of suitable results. Protocols may consist of medical treatment like chemotherapy, as well as radiotherapy, surgery, etc.

3.1 Task Structure and Natural Language Description

We can think of the generic task as a kind of plan with the associated PSM "*act_check_decide*", as is represented in figure 2.a. After the "act and check" of a step we have to decide the next step according to a set of reasons. This decision can be refined according to the inference structure of figure 2.b, making explicit the knowledge that was implicitly associated with the decision. We substitute the sequence *{check_complains→ decide}* by *{patient_state → select_1}*, *{plan_situations → select_2}* and *{compare}*.

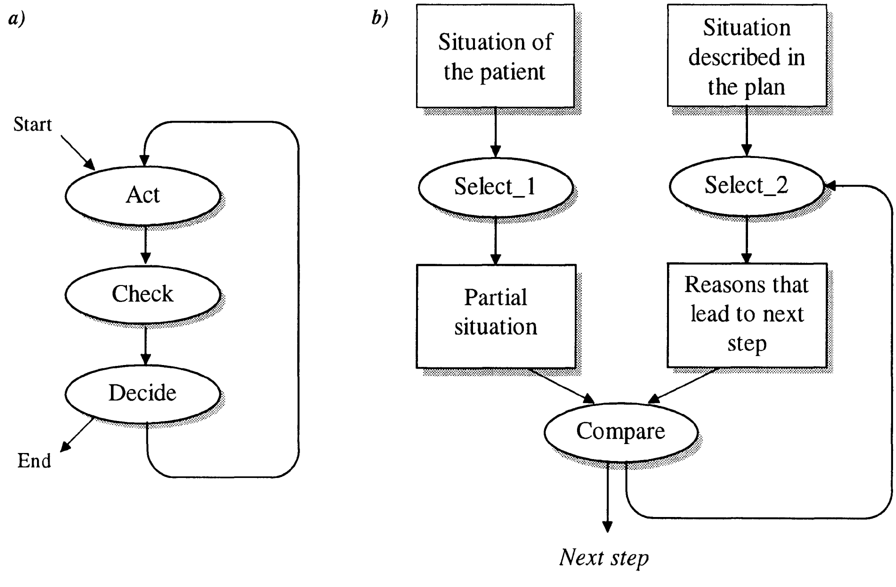


Fig 2. Generic task "to carry out a plan". a) PSM "act-check-decide". b) Refinement of the *{decide}* inference into *{select_1, select_2* and *compare}*.

Once we have a description for the generic task model, we can propose a suitable natural language description, as follows, for the steps of an hypothetical protocol, where depending on the state of the patient, different medicines (or any other treatments) are suitable:

Protocol "Protocol name".

Apply step "1".

After step "1" **we may have** "2", "3".

Apply step "2" **if state is** "(reasons ...)".

Apply step "3" **if state is** "(reasons ...)".

In step "1" **apply** "(medicines and doses...)".

etc...

The bold types represent the description related to generic task, while the rest represents domain knowledge. We impose that such a clear description like this expresses precisely what we mean, so there is not ambiguity at all.

3.2 The Hierarchic Graph Interpreted in Terms of Protocols

As to the expertise model, we can interpret **D** as the domain knowledge, while **Φ**, φ and Γ can be interpreted in terms of the generic task knowledge. As to the domain knowledge on protocols, we interpret the elements belonging to **A** as the *reasons* that lead us to the next step, the elements belonging to **S** as the consecutive *steps*, and the elements belonging to **M** as the *medicines* (or any other treatment) for the patient. By comparing this interpretation with the natural language description suggested before, we can build the following table.

Natural language statements	*Relationships with the graph elements*
Protocol "Protocol name".	**Protocol P_k .**
Apply step "1".	**Apply step S_1 .**
After step "1" **we may have** "2", "3".	**After step S_1 we may have S_2 , S_3 .**
Apply step "2" **if state is** "(reasons ...)".	**Apply step S_2 if state is A_2 .**
Apply step "3" **if state is** "(reasons ...)".	**Apply step S_3 if state is A_3 .**
In step "1" **apply** "(medicines and doses..)".	**In step S_1 apply M_1 .**

As to the generic task knowledge on protocols, we interpret:

Expression	*Interpretation*
$\mu(S_i \cup \mathbf{P_k}) = \mathbf{M_i} \cup \mathbf{P_k}$	On step $\mathbf{S_i}$ of protocol $\mathbf{P_k}$, the patient must have the treatment $\mathbf{M_i}$.
$\Phi(S_i \cup \mathbf{P_k}, n) = \mathbf{S_s} \cup \mathbf{P_k}$	After step $\mathbf{S_i}$, a choice is step $\mathbf{S_s}$.
$\Gamma(S_i \cup \mathbf{P_k}, n) = \mathbf{S_i} \cup \mathbf{P_k} \cup \mathbf{A_s}$	After step $\mathbf{S_i}$, some of the characteristics of the patient can be $\mathbf{A_s}$.
$\varphi(S_i \cup \mathbf{P_k} \cup \mathbf{A_s}) = \mathbf{S_s} \cup \mathbf{P_k}$	After step $\mathbf{S_i}$, if the patient characteristics are $\mathbf{A_s}$, apply step $\mathbf{S_s}$.

Based on a depth-first search throughout the whole hierarchic graph, we can describe an algorithm to generate the hierarchic graph structure from the natural language description. This algorithm depends on the generic task; however since this task represents the most complex case, other algorithms should be a similar or simplified case. It is clear this is a constructive procedure.

3.3 The hierarchic graph interpreted in terms of programming

Now that we have the relationships between graph and natural language, we need the relationships between the graph and the programming elements. We are going to show these relationships for a rule base system. Similar descriptions can be obtained for a 3rd generation language, languages like Common LISP, or programable logic implementations of the equivalent finite state automaton.

The case of a rule based system is quite easy, if we consider φ. Bear in mind that

$$\Phi(S_{\mathbf{i}} \cup P_{\mathbf{k}} \cup A_{\mathbf{s}}) = S_{\mathbf{s}} \cup P_{\mathbf{k}}$$

defines arranged pairs

$$R_{\mathbf{iks}} = (S_{\mathbf{i}} \cup P_{\mathbf{k}} \cup A_{\mathbf{s}}, S_{\mathbf{s}} \cup P_{\mathbf{k}})$$

and also

$$\mu(S_{\mathbf{u}} \cup P_{\mathbf{k}}) = M_{\mathbf{u}} \cup P_{\mathbf{k}}$$

defines arranged pairs

$$R^{\mu}_{\mathbf{uk}} = (S_{\mathbf{u}} \cup P_{\mathbf{k}}, M_{\mathbf{u}} \cup P_{\mathbf{k}}).$$

So $\mathbf{R_{iks}}$ can represent a rule, where $S_i \cup \mathbf{P_k} \cup \mathbf{A_s}$ is the antecedent and $\mathbf{S_s} \cup \mathbf{P_k}$ is the consequent and analogously for $\mathbf{R}^{\mu}_{uk}$. Disregarding $\mathbf{P_k}$ as constant, the elements belonging to S_i, S_s and $\mathbf{A_s}$ can be translated into programming elements, as usually in a KBS. For instance, suppose a simple hypothetical case, if $s_{ij} \in \mathbf{S_i}$ represents step "1", $s_{sj} \in \mathbf{S_s}$ represents step "2", $a_{sj} \in \mathbf{A_s}$ represents "temperature greater that 37 °C", and

$m_{Sj} \in \mathbf{M}_S$ represents "one aspirin", we can define, respectively, programming variables like "number-of-aspirines = 1" and programming expressions to evaluate like "temperature > 37". A similar procedure was proposed by Barreiro et al. [1], they call it *generalized magnitudes*. It is easy to see that also any of these expresions could be a predicate. On the other hand, the inference mechanism or inference engine is represented mainly by φ, but also by $\mathbf{\Phi}$ and $\mathbf{\Gamma}$.

Let us see an example for Nexpert Object. This environment has a language for defining rules and other elements, like objects, called text format. So we have to associate the elements describing our graph with Nexpert Object's text format elements. In order to implement $\mathbf{S}_i$, $\mathbf{A}_i$ and $\mathbf{M}_i$ as objects we associate the elements $s_{ix} \in \mathbf{S}_i$ with a form

$$[\, \alpha^{s}_{ix} \,(\, \beta^{s}_{ix} \,,\, \delta^{s}_{ix} \,)\,]$$

where

α^{s}_{ix} represents an assign operator

β^{s}_{ix} represents the destination operand

δ^{s}_{ix} represents another operand

we associate every $a_{iu} \in \mathbf{A}_i$ with

$$[\alpha^{a}_{iu} \,(\beta^{a}_{iu} \,,\, \delta^{a}_{iu} \,)\,]$$

where

α^{a}_{iu} represents a relational operand

β^{a}_{iu} represents an operand

δ^{a}_{iu} represents another operand,

and analogously we associate every $m_{iv} \in \mathbf{M}_i$ with

$$[\alpha^{m}_{iv} \,(\beta^{m}_{iv} \,,\, \delta^{m}_{iv} \,)\,]$$

where

α^{m}_{iv} represents an elemental operator (sum, assign, etc.)

β^{m}_{iv} represents the destination operand

δ^{m}_{iv} represents another operand.

Now we can associate these forms with Nexpert Object text format syntax expressions like

(operator (property-slot) (value))	in general
(operator (property-slot))	when operator is "Yes"

So, we associate

(Yes (cm_{iv}))

with s_{ix} ,

(oa_{iu} (ca_{iu}) (va_{iu}))

with a_{iu} , and

(omiv (cmiv) (vmiv))

(Yes (cmiv))

with m_{iv} , where

oaiu , omiv ,Yes	are operators,
csix , caiu , cmiv	are property-slots,
vsix , vaiu , vaiv	are values.

Then for the $\mathbf{R}^{\mu}_{uk}$ rules, we can write:

```
(@RULE= U
        (@LHS=
                ( Yes ( $cs_i$ ) )
        )
        (@HYPO= $cs_i$ )
        (@RHS=
                ( $om_{i1}$ ( $cm_{i1}$ ) ( $vm_{i1}$ ) )
                ( $om_{i2}$ ( $cm_{i2}$ ) ( $vm_{i2}$ ) )
                ( $om_{i3}$ ( $cm_{i3}$ ) ( $vm_{i3}$ ) )
                ...
                ( $om_{in}$ ( $cm_{in}$ ) ( $vm_{in}$ ) )
        )
)
```

and for the $\mathbf{R}_{iks}$ rules, we can write

```
(@RULE= X
        (@LHS=
                ( Yes ( $cs_i$ ) )
                ( Yes ( $cp_k$ ) )
                ( $oa_{i1}$ ( $ca_{i1}$ ) ( $va_{i1}$ ) )
                ( $oa_{i2}$ ( $ca_{i2}$ ) ( $va_{i2}$ ) )
                ( $oa_{i3}$ ( $ca_{i3}$ ) ( $va_{i3}$ ) )
                ...
                ( $oa_{in}$ ( $ca_{in}$ ) ( $va_{in}$ ) )
        )
        (@HYPO= $cs_u$ )
)
```

Now, the relationships between the natural language statements and the programming code we choose are clear, since we established the relationships between natural language and the hierarchic graph, and we just established the relationships between the hierarchic graph and the programming code.

4 Hierarchical Classification as a Basic Diagnosis Task

Our long term conjeture is that three task structures: one of analysis (to make a diagnosis), another of synthesis (to propone and follow a plan) and a third of modification of the other two in accordance with the expert's experience (modification of diagnosis and therapy plan parameters by learning) may be sufficient to model the basis aspects of reasoning in many domains [8].

In section 3 we have considered the generic task of "to carry out a plan". Here we present briefly the hierarchical classification as a basic task in medical diagnosis. According to the classic Chandrasekaran's description of hierarchical classification [3], where the method was described in terms of "establish" and "refine", each "establish" could be associated with a node, if we think of the nodes as different degrees of refinement, these degrees of refinement being states (and the permitted inferences being the permitted refinements). Note that similar descriptions can be found in KADS' library of generic tasks [12] and as part of Clancey's heuristic classification [4]. Therefore, we could achieve a description like

> ...
>
> Refine acute lymphocitic leukemia into L1, L2, L3.
>
> Establish L1 as 75% homogeneous lymphocite population, small lymphocites, with scarce cytoplasm, relatively coarse cromatin pattern, regular nuclear shape, conspicuous nucleoli in more than 75% of them.
>
> ...

from the FAB classification of leukemias. By means of the algorithm we can generate the graph elements. The algorithm for the hierarchical classification is a simplified case with regard to the most complete case of the generic task to carry out a plan. Due to the structure of the natural language statements, we can represent their elements by symbols:

Symbolic expression	*Represents...*
$(\varepsilon^{E}_{i}, \varepsilon^{1}_{i}, \varepsilon^{A}_{i}, \varepsilon^{2}_{i})$	an "establish" statement
$(\rho^{R}_{i}, \rho^{1}_{i}, \rho^{I}_{i}, \rho^{2}_{i1}, \rho^{2}_{i2}, \ldots \rho^{2}_{in})$	a "refine" statement

where

Algorithm's symbol	*Knowledge level element*	*Graph element*
ε^{E}_{i}	the word "Establish"	φ
ε^{1}_{i}	the concept to establish	S_i
ε^{A}_{i}	the word "as"	none
ε^{2}_{i}	the characteristics	A_i
ρ^{R}_{i}	the word "Refine"	Φ
ρ^{1}_{i}	the concept already established	S_i
ρ^{I}_{i}	the word "into"	none
$\rho^{2}_{i1}, \rho^{2}_{i2} \ldots$	the possible refinements	$S_k, S_m, \ldots$
lower index *i*	an arbitrary statement number	none

In order to associate a meaning with the set $\mathbf{P}_k$, we introduce (π^{c}_{k} , π^{1}_{k}), where π^{1}_{k} is related to such set. Then, the algorithm can be described as follows (note that $S_0 = \varnothing$):

```
{Algorithm that generates the graph structure from the natural language statements}
Begin.
    Search $\varepsilon^1_i$ such that there is not any $\rho^2_{sk}$ such that $\rho^2_{sk} = \varepsilon^1_i$ .
    When you find it, let $\varepsilon^1_i = S_1$ and then let $\varepsilon^2_i = A_1$; so we have built
    $\varphi(\mathbf{P}_k \cup \mathbf{A}_1) = S_1 \cup \mathbf{P}_k$ .
    Depth ( $\varepsilon^1_i$ )
End.
```

And the procedure "Depth" can be described as follows:

```
{Depth ( $\lambda$ )}
Begin.
    Search $\rho^1_k$ such that $\rho^1_k = \lambda$.
    When you find it, for every $r$ in the same statement $k$ (where $\rho^1_k$ was found) do:
    Begin.
        Let $\Phi( \rho^1_k \cup \pi^1_k , r ) = \rho^2_{kr} \cup \pi^1_k$ .
        Search $\varepsilon^1_s$ such that $\varepsilon^1_s = \rho^2_{kr}$ .
        When you find it, take $\varepsilon^2_s$ in the same statement $s$ and
        let $\Gamma( \lambda \cup \pi^1_k , r ) = \lambda \cup \pi^1_k \cup \varepsilon^2_s$ ,
        let $\varphi( \lambda \cup \pi^1_k \cup \varepsilon^2_s ) = \varepsilon^1_s \cup \pi^1_k$ .
        Depth ( $\varepsilon^1_s$ ).
    End.
End.
```

By means of this algorithm, we can find out the underlying computational structure for the problem (stuck to these statements only), as we can see in figure 3.

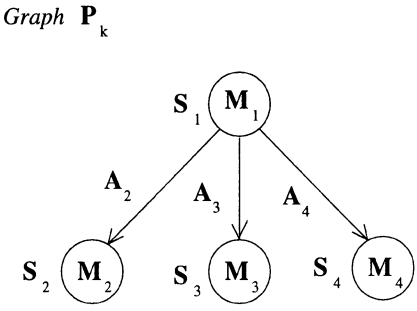


Fig 3. An easy example for the FAB classification of leukemias.

And following the method we explained before, we could write the following rule:

```
(@RULE= RES005
     (@LHS=
          (Yes   (leukemia.lymphocitic_acute))
          (>     (lymphocites.homogeneous_population_percentage)    (75.0))
          (=     (lymphocites.size)    ("SMALL"))
          (=     (lymphocites.cytoplasm_volume) ("SCARCE"))
          (=     (lymphocites.cromatin_pattern)   ("COARSE"))
          (=     (lymphocites.nuclear_shape)    ("REGULAR"))
          (>     (lymphocites.percentage_with_conspicuous_nucleoli)  (75.0))
     )
     (@HYPO= leukemia.L1)
)
```

where the interpretations we made are easy to understand, since we used mnemonics for the names of the property slots and values, besides we provided the natural language description; but it is clear there is an implicit translation table between natural language statements and hierarchic graph elements, like

element	*set*	*meaning*
s_1	S_1	acute lymphocitic leukemia
s_2	S_2	L1
s_3	S_3	L2
s_4	S_4	L3
a_{21}	A_2	75% homogeneous lymphocite population
a_{22}		small lymphocites
a_{23}		with scarce cytoplasm
a_{24}		relatively coarse cromatin pattern
a_{25}		regular nuclear shape
a_{26}		conspicuous nucleoli in more than 75% of them

and an analogous table between hierarchic graph elements and text format elements, we do not provide. One have to bear in mind these tables. Note that for a hierarchical classification, the hierarchic graph has not any cycle; note also that the interpretation of the elements of the graph in terms of the domain knowledge depends, of course, on the generic task; however the graph elements that are interpreted as domain knowledge, inferences, etc., are always interpreted like this for all generic tasks.

5 Conclusions

Whether one accepts our formalism or whether one does not, the explicit transition between natural language and program code is always needed [13]. This means there is an interpretation of either the formalism or the program code, that must be made in terms of natural language semantics *or else* we could never understand what the formalism means and what the program code means. Once acepted that the meaning of the models of expertise always remain in the external observer domain and at the knowledge level, the rest of our proporsal in this article is crystal clear: there is a common structure underlying a representative set of generic tasks and PSM's, if we think of generic tasks in terms of states of knowledge, and then we represent these states (and the corresponding state transition conditions) by means of a hierarchic graph. In these cases, a constructive method to obtain the program code can be described, if we express the generic task model by means of natural language. Mainly domain knowledge will determine the terms to interpret the state (for instance, we can interpret a state as the degree of refinement in a hierarchical classification). Following the method we have shown in this paper, we can establish relationships between a natural language description of a generic task and a hierarchic graph; this relationship is constant given the generic task. Since the relationships between the hierarchic graph and the programming code are constant given the program language, we can describe the method to generate the hierarchic graph structure from the natural language statements, and automatically generate the program code, given the generic task and the program language.

Therefore, if the formalism included in our *reduction method* is a powerful tool for avoiding the lack of consistency and precision of conceptual models, it is because of the *clear*, *precise* and *unequivocal translation tables* (meanings and causalities) we use for first reducting and later interpreting such formalism. These semantic tables tell us about the richness of what *we* mean when *we* wrote the formalism.

Acknowledgements

We acknowledge the financial support of the Spanish CICYT under projects TIC-94-95, and TIC- 97-0604.

References

1. Barreiro A., Otero R.P., Marín R. & Mira J.: A modular Knowledge Base for the Follow-Up of Clinical Protocols. *Methods of information in Medicine* **Vol. 32. No. 5** (1993) 373-381.
2. Benjamins V.R. & Aben M.: Structure-preserving knowledge-based system development through reusable libraries: a case study in diagnosis. *International Journal of Human-Computer Studies*, **47** (1997) 259-288.
3. Chandrasekaran B., Johnston T.R., & Smith J.W. Task Structure Analysis for Knowledge Modeling. *Communications of the ACM*, **35, (9)** (1992) 124-136.
4. Clancey W. J.: Heuristic Classification. *Artificial Intelligence* **27 (3)** (1985) 289-350.
5. Mira J., Delgado A.E., Boticario J.G. & Díez F.J. *Aspectos básicos de la inteligencia artificial.* Sanz y Torres (1995*a*).
6. Mira J. & Delgado A.E. Reverse Neurophysiology: the "Embodiments of Mind" Revisited. *Proceedings of the International Conference on Brain Processes, Theories and Models.* R. Moreno & J. Mira, Eds. **37-49** (1995*b*). Cambridge, MA. The MIT Press.
7. Mira J., Herrero J.C. & Delgado A.E. (1997a). A Generic Formulation of Neural Nets as a Model of Parallel and Self-Programming Computation. *Biological and Artificial Computation: From Neuroscience to Technology.* 195-206. J. Mira, R. Moreno-Díaz & J. Cabestany, Eds. Springer-Verlag.
8. Mira J., Rives J., Delgado A.E. & Martínez R.: Towards the Unification of the Inference Structures in Medical Diagnosis Tasks. *Methods of Information in Medicine* (1997b). Accepted for publication.
9. Newell A.: The Knowledge Level. *AI Magazine* **2 (2)** (1981). 1-20, 33.
10. Nilsson N.J.: *Principles of Artificial Intelligence.* Tioga Publishing. (1981).
11. Speel P. & Aben M.: Applying a library of problem-solving methods on a real-life task. *International Journal of Human-Computer Studies*, **46** (1997) 627-652.
12. Tansley D.S.W. & Hayball C.C.: *Knowledge-Based Systems Analysis and Design. A KADS Developper's Handbook.* Prentice-Hall (1993).
13. Turing A.M. On computable numbers. *Proceedings of London Mathematical Society*, **42** (1937).

14. Van Harmelen F. & Aben M.: Structure-preserving specification languages for knowledge-based systems. *International Journal of Human-Computer Studies*, **44** (1996) 187-212.

15. Wielinga B.J., Acreiber A.T. & Breuker J.A.: KADS: A Modelling Approach to Knowledge Engineering. Knowledge Acquisition **4** (1992) 5-53.

16. Winston P.H.: *Artificial Intelligence, Third Edition*. Addison Wesley (1992).

Theory of Constructible Domains for Robotics: Why?

Marta Fraňová, Monia Kooli

CNRS & Université Paris Sud, LRI, Bât. 490, 91405 Orsay, France
mf@lri.lri.fr, monia@lri.lri.fr

Abstract. The goal of *Recursion Manipulation* (*RM*) is to design a *calculator* that provides formal proofs for a particular type of formulae corresponding to the task of program construction and program verification of *recursive* procedures. Recalling first that Gödel's result *cannot* be used as a mathematically justifiable argument *against* RM, the paper illustrates the strategic importance of *RM* in the design of autonomous, self-reprogrammable robots. In particular, on the basis of more technical papers making a necessary theoretical support for this paper, we illustrate that for roboticians it is *sufficient* to be concerned with an *external* characterization of *RM*. Relying on the *Theory of Constructible Domains*, the framework of *RM* is powerful enough to take care of logical justifications inherent to various forms of induction schemes (i.e., the termination justifications for recursive programs and plans). The paper illustrates also that two, not necessarily compatible, types of efficiency are related to recursive plans.

1 Introduction

One of necessary features of a *user-independent* system is the possibility of opaqueness of the system's internal mechanisms for a user using it for standard purposes. Thus, an automated system is not always user-independent. A design of a robot working in an environment requiring a user-independence of the robot necessarily requires an understanding of the essential difference between an automated and a user-independent system. Fully understanding the potential of the user-independence of a system is then just a small mental step away from somewhat interesting conclusions[1].

The **goal** of Recursion Manipulation (*RM*) is to design a ***user-independent*** system able to construct and verify recursive programs and plans which are specified formally by the essential property that holds in the considered environment between the input arguments and the output once this plan is applied to its input arguments. Since some forms of *inductive reasoning* (i.e., 'creative') are necessarily integrated into the deductively oriented *RM*, as we recall in [5], an *RM* system *is not* in contradiction with Gödel's results — it should be (or become) clear that these are not related to a deductive environment enhanced by *inductive* tools. Thus, the notion of undecidability is *irrelevant* to *RM*. This *must not* be interpreted as taking undecidable problems and making them decidable. It has to be understood, for problems solvable 'by-hand', as a *rigorous*, i.e., mathematically justifiable, and *practically manipulable* organization of 'infinite chaos' (which is a sign, a symptom, of undecidability) so that appropriate inductive tools (relying on the rigor and routine

[1] For instance, when a robot is accidentally damaged while the core of its logical part remains intact, if this robot is able to construct and verify programs (and plans) user-independently, it will be able to reprogram its own damaged parts in the best case and, if this is not materially possible, it may adapt itself to a new role, a role for which it was not initially designed but which is also not incompatible with the original purpose and the environment in which it finds itself. With respect to the high cost of robots, this self-adaptation may be of some economic value.

character of the deductive framework to prepare a kind of 'organized infinity') can be well-specified, developed for, and used in a deductive framework without the drawback of bringing a logical inconsistency of the resulting extended framework.

Constructive Matching Methodology (*CMM*) represents one, and so far unique, approach[2] trying to achieve the goal of *RM*. Since *CMM* builds a user-independent system, roboticians (or ordinary programmers) *do not need to know how it works*. They only need to know *what to use it for*. Nevertheless, the exact specification of a *standard* use of an *RM* system contains an implicit requirement: in order to construct and verify recursive programs and plans user-independently, some particular formalization of the intended domain *has to be given*. This is not and cannot be the task of the experts in *RM*. On one side, the experts in *RM* cannot foresee all the domains that may be interesting for roboticians or ordinary programmers. Often, by their confidential character, these domains may even be (and/or should remain) semantically 'inaccessible' to the experts in *RM*. In consequence, the following can be interesting information to be aware of:

A positive result of *CMM* is that it shows that, for an intended domain, *it is sufficient to work out such a formalization only once*. *CMM* is able to take care of possible reformulations that might arise when a system is used by the users that did not worked out this initial formulation. This is one of the points we shall illustrate in this paper, since, as it will become clear, it is of a great importance for understanding the consequences (and *necessary* characteristics) of the user-independence of an *RM* system. The second positive feature is that *RM* experts *do not need to be involved in the process of the formalization*. In other words, among roboticians, only a small group should devote themselves to working out an initial formalization for any new intended domain of interest. We shall call *domain expert* any person who is not an expert in *RM* and who has to formalize an intended domain for the purpose of the user-independent construction and verification of recursive programs and plans. A domain expert may feel uneasy[3] with the kind of formalization that is related to recursion. By its *Theory of Constructible Domains* (*Theory of CDs*), which is a particular theory of recursive programs developed for and used in *CMM*, *CMM* does not guide the domain experts 'to get the induction schemes right', but to give the system *adequate information* that will allow the system 'to be autonomous in getting induction schemes right' as well *adequate information* that will enhance the system's logical power. The first type of information concerns an inductive definition for the intended domain. It is not surprising that a requirement for this kind of information is necessary, since, whenever a recursive computation is justified — terminating — there is, somewhere, some kind of inductively defined system of objects. The second type of information, as explained in [4], concerns the domain's *semantic* of and thus it cannot be a part of *Theory of CDs*, which is, in its abstract version, a formal theory. However, *CMM* is built so that it is aware of the character of *semantic aspects related to Recursion Manipulation* and that can and *must* be made explicit in order to enhance the logical power of an *RM* mechanism. In consequence, the formalization 'effort' of the domain experts consists of using *their own domain expertise*, *their own skills*, rather than in working out mathematical skills for a (mathematical) formalization of meta-theoretical *and* methodological *and* epistemological arguments.

Everything said above is simply a summary of an *external* behavior (or appearance) of the results achieved in *CMM* so far. [2] can be considered as a somewhat detailed presentation (necessarily so) of the *CMM*'s internal mechanisms in

[2] The 'skeleton' of the epistemologically justifiable mathematical model for *CMM* was designed in 1983-1984, and is implicitly present in [1]. A complete list of the publications on *CMM*, on experiments illustrating the power of *CMM*, as well as on the experimental implementation of the system *PRECOMAS* (*PRoofs Educed by COnstructive MAtching for Synthesis*) can be consulted via http://www.lri.fr/Francais/Recherche/ia/ (or http://www.lri.fr/~mf).

[3] For instance, see in [15] the acknowledgment for « getting the induction schemes right ».

the process of solving a rather simple planning problem. The present paper is a part of a series devoted to an easy-to-follow ***illustration*** of *CMM* in the framework of the user-independent construction and verification of recursive *plans*. The first paper of this series, namely, [5] focuses on illustrating the unity of theoretical and practical description of *CMM* as a *theorem proving mechanism*, it also illustrates the non-existence of the well-known frame-problem in a system relying on this particular theorem proving mechanism. The present paper focuses on illustrating the unity of the theoretical and practical description of *CMM* as a *recursion justification mechanism*. Namely, we use a very simple example to illustrate the unity of the theory and computation required by *RM* for justifying recursion, i.e., the induction principle and defining by recursion. The choice of the example presented in section 4 was motivated by the following reasons:

(a) it allows us to illustrate a domain expert's 'duties' during the process of formalizing of two simple intended worlds in which recursive plans are meaningful;
(b) it allows us to illustrate that *RM* has to deal with two, not necessarily compatible, types of efficiency that recursive programs have;
(c) it allows us to illustrate that, although multidisciplinary, *RM* has its *own place* as a proper scientific field with a well-specified goal (specified above) and with its own appropriate methods of development (illustrated 'statically' by *CMM*'s *recursive* conceptual architecture which is rendered unique and inimitable by making the presence of a particular *inductive* reasoning mechanism *externally invisible*).

The next section provides more feedback for (a) and (c) and completes the above list of the reasons. The development of the example presented in section 4 relies on the above mentioned *Theory of Constructible Domains*. This theory is very briefly described in section 3.

2 A Note On 'Related Work'

Kunen, in [8], uses the system developed by Boyer-Moore, known as Nqthm in the literature, to verify the Paris-Harrington version of Ramsey's theorem. The proof [4] presented by Kunen relies on ε_0 induction, thus illustrating that « ... Nqthm allows definition of functions by recursion on the ordinal ε_0 , and proofs by induction on ε_0 ... ». As also mentioned in [18], pg. 705, transfinite induction is, in its extreme form, « ... irreconcilable with constructivity ... ». In consequence, as far as recursive programs and plans are concerned, they necessarily correspond to inductions (i.e., the recursions of these recursive programs are logically justified by inductions) that are *constructive* in a computational sense. If ε_0 is *necessary* for defining a function by recursion, (i.e., if it is not just a *matter of convenience* as mathematicians are used to,) then the corresponding recursive definition of this function shall not correspond to a recursive computationally manipulable program.

Thus, the first conclusion that is important for roboticians is: *RM* does not need to be preoccupied by induction on ε_0. *RM* does not need to use the full Nqthm's theoretical background. On the other hand, it is natural to suppose that, since such a powerful induction, as ε_0 induction, is allowed by a system, then ordinary, simple, (i.e., constructive forms of inductions) are allowed as well. This supposition is natural, but *erroneous*, as the following example illustrates.

Let us have a look at the following definition of the function segment (pg. 230 in [8]):

```
(defn segment (m n)
      ( if     (and (leq m n) (numberp m) (numberp n))  (cons m (segment (add1 m) n) nil )
      ( ; hint
               (lessp (difference (add1 n) m) ...) ) )
```

The first part of this lisp program is logically sufficient to define the function segment. The second 'hint' part, is given by Kunen to make Nqthm accept this program as a justified

[4] We shall leave it for concerned mathematicians to appreciate the virtuosity by which Kunen constructs, by hand, a scenario for a proof of Ramsey's theorem in a way that makes it possible to run this scenario by Nqthm which was developed for verifying purely universally quantified formulae. Kunen's scenario relies on several formulae that contain existential quantifiers (as does the Paris-Harrington version of Ramsey's theorem).

definition, i.e., a terminating program. In other words, Nqthm is *unable* to perform such a logical justification itself; Nqthm does not allow the recursion of this program unless a user orders it to accept it. This illustrates a discrepancy, a gap, a lack of coordination in the development of 'theory' and the development of 'practice'. In fact, for *user-dependent* systems, such a lack is inessential: the presence of an ingenious user is assumed (required and guaranteed). But, as far as a rigorous justification of the *user-independence* of an *RM* system is concerned, it illustrates that, in a sense, not all the forms of 'constructive' inductions are 'allowed' — in practice — by theoretical approaches to 'implementing induction' [7], [11], [12], [13], or even higher order considerations [14] and well-known mathematical theories of recursion. This illustrates that *RM* needs its own 'theory of constructive induction', a theory *manipulable* with respect to *RM* 's algorithms.

To use the language of programmers, as explained in [4], 'constructive' induction is closely related to termination of recursive programs (it corresponds to it, in fact). Thus, it might seem that to automatically 'allow' the above program in Nqthm, it is sufficient to use some termination verification algorithm developed in computer science ([6], [19]). As explained in [4], such a suggestion does not provide a general solution in the frameworks for which the undecidability aspects are relevant — in fact, all the user-dependent approaches to automated program construction and verification seem to use this notion in order to justify the need for a user. As a consequence of this situation (the mentioned 'relevancy' of undecidability and a lack of a rigorous, manipulable, description of the exact logical and manipulation power of the termination-verifiers), somewhat surprising solution is accepted in automated inductive theorem proving: in [10], the authors suggest that the programmers should learn to program in the style falling under « Walther's recursion » (see in [19]). Thus, let us push further our illustration. Since the Kunen's program segment above does not fall under Walther's recursion, if Nqthm is extended by Walther's algorithm, Kunen just needs to reimplement his program in Walhter's style and then no hints are needed to give to Nqthm. (Since Nqthm is 'sensitive' to the form of the given definitions, shall Nqthm run Kunen's scenario also with this new form?) In section 4 we will show a somewhat unexpected *practical* consequence of this rather unnatural requirement. (Are machines constructed to facilitate the work of humans or do *people* have to facilitate the work of machines?)

True roboticians might now argue that the above segment function concerning *natural numbers* is in no relation to *real* world problems that a robot may be intended to solve. The rest of this article is thus also devoted to clearing up this possible misunderstanding. In other words, we claim that if an *RM* system were unable to solve the problems related to the construction and verification of recursive programs for natural numbers, such a system would be unsuitable for real world problems as well. Thus, in addition to (a), (b) and (c) of the previous section, the example presented in section 4:

(d) illustrates that a recursive 'plan' can be formulated which is almost 'identical' to Kunen's program for the segment function;

(e) explains a pragmatic meaning of the restrictions imposed by Walther's recursion in the framework of recursive planning.

3 A Note On *Theory of Constructible Domains*

Recall that *Recursion Manipulation* is concerned exclusively by the construction and verification of recursive, (i.e., computational) programs and plans. *RM*, namely its user-independence, requires a mathematically rigorous framework, in which 'defining a procedure by recursion', for programs, is a logically justified definition formation rule and in which all the various recursion forms of programs can logically be justified. In consequence, the goal of *RM* implicitly contains the task of giving a logically justifiable feedback for a non-standard verification of the termination of programs. *Constructive Matching Methodology* does this via the *Theory of CDs* [5] and uses *RM* (thus, *CMM*) itself in establishing such a manipulable theoretical framework. The *Theory of CDs* represents a new reformulation of the *elementary* theory of recursive

[5] [3] *and* [4] represent a full *technical* description of the *Theory of CDs.*

functions (i.e., all the recursive functions that can be given as computable programs). The most important achievements of the *Theory of CDs*, that are related to *RM*, are as follows (if necessary, see terminology in [3]):

♦ the formulation of a particular world as a Constructible Domain provides and justifies logically (independently of an expert) *standard* and, whenever appropriate, *usual* induction principle schemes;

♦ in the framework of *CMM*, the *non-usual* induction principle 'schemes', whenever they correspond to terminating programs, are logically justified as well and the recursive definitions relying on these non-usual recursions can, whenever appropriate, be user-independently reformulated, in the framework of *CMM*, into their usual-recursion versions.

4 Example

[5] recalls that, to use *CMM* for the construction and verification of recursive plans, *CMM* adopts (and somewhat extends, [2]) the *plan theory* developed by Manna and Waldinger in [9]. This *mw-plan theory*, as we call it, seems perfect to us for extending the application field of *RM*, initially thought of for recursive programs construction and verification. However, even though the mw-plan theory is illustrated (in [9]) for *recursive* programs, it is independent of recursion theories. Thus, as far as *RM* is concerned, the first contribution of the mw-plan theory is that it renders it possible to consider the recursion aspects of recursive *plans* within the theoretical framework of a *classical logic* recursion theory. The second contribution of the mw-plan theory is that it hints at a meaningful way to construct plans within the formal framework of the mw-plan theory (i.e., within a non-classic logic) *via* a classical logic theorem proving *mechanism* [6]. The first contribution explains why we feel (and we actually are) authorized to limit our illustration here to simple classical logic worlds, namely two worlds that can be represented by simple inductive structures and that are nevertheless interesting for 'moving-in' robots[7].

In this section, our first goal is to give an initial formalization to the world of blocks (on a table, in a room) specified in [9]. These blocks are all the same size, so that only one block can fit directly on top of another (this restriction was formulated by Manna and Waldinger, but it is not essential for *Theory of CDs*, provided the expert domain specifies this formally). We call *AB* (standing for *ALL_BLOCKS* in [2]) the set of all these blocks. If no other block is on top of a block, Manna and Waldinger characterize it as *clear*. We shall stick to this terminology here.

Intuitively speaking, to describe a world as a constructible domain, the expert of this world must know how to define it inductively first, more or less informally. For instance, let us consider the following more or less informal inductive definition.

○ if block x from *AB* is clear, then x is an element of *FAB* ;

○ if block x from *FAB* is not on table then the block that is just below x belongs to *FAB* ;

○ any element of *FAB* can be obtained by the previous two rules in a finite number of steps.

Thus, the set *AB*, which initially has no particular structure is now looked at via an inductively defined system of objects, namely the system *FAB* (standing for 'formalized *AB*'). Of course, the informal character of this 'definition' is evident (by reference of *FAB* to *AB*). Thus, from this informal definition it is necessary to extract a *formal* inductive definition by explicitly expressing the constructors of the intended domain's formal

[6] As we point out in [5], it seems to us that the value of the mw-plan theory for practical purposes can be slightly concealed by Manna and Waldinger's use (in [9]) of a user-dependent environment for recursive plans construction. We remedy this in [2] and [5], where we place their theory into a user-independent environment, namely, that of *CMM* methodology. Thus the practical value of their plan theory becomes evident.

[7] In other words, what is illustrated here for a classical logic description of a world holds for its situational variant, formalized of course, in terms of the plan theory developed by Manna and Waldinger (or its logical equivalent).

description which is formally independent of $\mathcal{FAB}$ (even though the semantic dependence shall be present implicitly, namely, $\mathcal{AB}$ is a model for $\mathcal{FAB}$.)

We shall leave out of the technical side of this transformation for our purposes here. ([4] is an introductory reading, [3] contains more technical details and justifications, and [2] presents the details formulated in the terminology of this example.) We can thus suppose that the expert (a robotician) shall come out with a formalization of the initially unstructured world of blocks that is very similar to the formalization presented below.

4.1 The role of the *Theory of Constructible Domains* in a formalization of $\mathcal{FAB}$

We shall call *initial block* each block from $\mathcal{AB}$ which is clear, i.e., no other block is on the top of this block. A formal counterpart to this semantic description is the following abstract set and predicate. Below, the marked paragraphs correspond to the contribution of the domain expert.

Let $\mathcal{W}$ be a set of elements. Let $\mathcal{IB}$ (standing for $\mathcal{INITIAL\_BLOCKS}$) be a (dynamically extensible) finite subset of $\mathcal{W}$. Let the predicate clear $\mathcal{W} \to \mathcal{BOOL}$ be defined by

$$\mathrm{clear}(x) \Leftrightarrow x \in \mathcal{IB}.$$

The notion of a 'dynamically extensible' set is related to the fact that a robot moving in the world of blocks shall probably modify the position of blocks, thus a block that is clear in one particular moment may be 'not clear' in another moment. This of course shall not modify the instantaneous *inductive structure* of the intended world, it simply means that considering various sets of initial blocks (and thus appropriate 'versions' of the system $\mathcal{FAB}$) is possible. In consequence, for illustration purposes we can abstract from this dynamical character of $\mathcal{IB}$.

Let table be a constant element in $\mathcal{W}$ and such that table $\notin \mathcal{IB}$, let on: $\mathcal{W} \times \mathcal{W} \to \mathcal{BOOL}$ be a primitive binary predicate verifying the following set of properties: { not(on(x,x)), not(on(table,y)) } for any x and any y.

To specify a Constructible Domain, by Definition D.2.1 in [3], one needs to give the constructors of this domain. Thus, let us suppose that a domain expert specifies the constructors of $\mathcal{FAB}$ as follows:

$$\begin{array}{lccc} \textbf{initial\_block}: & \mathcal{IB} & \to & \mathcal{FAB} \\ \textbf{just\_below}: & \mathcal{FAB} & \to & \mathcal{FAB} \end{array}$$

The conditions of applicability of these constructors are given by the domain expert too:

$$\mathrm{Poss}(\textbf{initial\_block}(x)) \quad \Leftrightarrow \quad \mathrm{clear}(x),$$

for any $x \in \mathcal{W}$, and, for any $y \in \mathcal{W}$

$$\mathrm{Poss}(\textbf{just\_below}(y)) \quad \Leftrightarrow \quad \mathrm{not}(\mathrm{on}(y,\mathrm{table})).$$

Here **initial_block** corresponds to the intuitive interpretation that an initial block is a block, and the general constructor **just_below** corresponds to the intuitive interpretation that if x is a block that has no direct contact with the table, then the block **just_below**(x) is the block that is just below x. We specify — as an expert of the intended world should do with respect to the semantic of Manna and Waldinger's intended model — the uniqueness arguments axiom for the constructor **just_below**

$$\textbf{just\_below}(x) = \textbf{just\_below}(y) \Rightarrow x = y \qquad (4.1)$$

Therefore, by the *Theory of CDs*, there is a selector corresponding to this constructor, in correspondence to [9] that we name **hat**. By the *Theory of CDs*, this selector is specified by the equation { **hat**(**just_below**(x)) = x } and the application condition for **hat** is { Poss(**hat**(x)) $\Leftrightarrow$ not(clear(x)) }. By the *Theory of CDs*, in the well-founded relation $\angle$ induced by the constructors (and, in the *Theory of CDs* defined by recursion in natural numbers! — see the last paragraph of this section), whenever Poss(**hat**(x)) and Poss(**just_below**(y)) are verified, the following two properties hold: { y $\angle$ **just_below**(y) , **hat**(x) $\angle$ x }. The *Theory of CDs* gives now a constructor, as well as its equivalent selector version of the *standard* induction principle for $\mathcal{FAB}$. This selector version — interesting for our purposes here is the following:

$$\frac{\begin{array}{ll} A(a), & \text{if clear(a)} \\ A(\textbf{hat}(a)) \Rightarrow A(a), & \text{if not(clear(a))} \end{array}}{\forall x\, A(x)}$$

Moreover, the *Theory of CDs* justifies, for $\mathcal{FAB}$, the defining functions by *standard recursion*. This means that, whenever u0 is from $\mathcal{W}$ and h is a function of the appropriate domain and codomain, any system of equations of the form,

$$\begin{array}{lcll} g(x) & = & u0, & \text{if } x \in \mathcal{FAB} \,\&\, clear(x) \\ g(x) & = & h(\mathbf{hat}(x),g(\mathbf{hat}(x))), & \text{if } x \in \mathcal{FAB} \,\&\, not(clear(x)) \end{array} \quad (4.2)$$

or, equivalently,

$$\begin{array}{rcll} g(\mathbf{initial\_block}(x)) & = & u0, & \text{if } x \in \mathcal{FAB} \,\&\, clear(x) \\ g(\mathbf{just\_below}(y)) & = & h(y,g(y)), & \text{if } y \in \mathcal{FAB} \,\&\, not(clear(y)) \end{array} \quad (4.3)$$

with the unknown variable g, defines one, unique, function g: $\mathcal{FAB} \to \mathcal{W}$. For instance, a simple syntactic verification is sufficient to check that the program makeclear presented in [9] or in [2] is a definition of a function, (i.e., it is a terminating program).

Moreover, the *Theory of CDs* also brings forward a definition recursive (in $\mathcal{FAB}$) for the above mentioned well-founded relation $\angle$ induced by the constructors of $\mathcal{FAB}$. As clear from [3], this well-founded relation is not initially defined recursively in $\mathcal{FAB}$, thus this computational (in $\mathcal{FAB}$) reformulation (from natural numbers) is a very important achievement of the *Theory of CDs*. This new formulation is used for computations related to the extension of standard recursion to usual recursion as well as in the justification proofs related to non-usual recursion [4].

4.2 The role of *Recursion Manipulation* in checking termination of recursive programs

In order to give an example of a non-usual recursive definition of a function with the domain arguments in $\mathcal{FAB}$, we shall introduce another constructible domain.

Since $\mathcal{FAB}$ is now defined, we can define another constructible domain in terms of $\mathcal{FAB}$. Let us call it $\mathcal{STACK}$. Let the constructors of $\mathcal{STACK}$ be as follows:

$$\begin{array}{lccc} \mathbf{st\_nil}: & & \to & \mathcal{STACK} \\ \mathbf{st\_cons}: & \mathcal{FAB} \times \mathcal{STACK} & \to & \mathcal{STACK} \end{array}$$

Let the uniqueness arguments axiom for the constructor **st_cons** be as follows:

$$\mathbf{st\_cons}(x,u) = \mathbf{st\_cons}(y,v) \Rightarrow x = y \,\&\, u = v. \quad (4.4)$$

For $\mathcal{STACK}$ defining functions by *standard recursion* is justified by the *Theory of CDs*. This means that, whenever a0 is from $\mathcal{W}$ and r is a function of the appropriate domain and codomain, if u $\in \mathcal{FAB}$ & v $\in \mathcal{STACK}$ any system of equations of the form

$$\begin{array}{rcl} f(\mathbf{st\_nil}) & = & a0 \\ f(\mathbf{st\_cons}(u,v)) & = & r(u,v,f(v)) \end{array} \quad (4.5)$$

with the unknown variable f defines one unique function f: $\mathcal{STACK} \to \mathcal{W}$. Defining by standard recursion of n-ary functions is justified by *Theory of CDs* too. Thus, for instance, the following system of equations defines a recursive function

$$\begin{array}{rcl} st\_append(\mathbf{st\_nil},u) & = & u \\ st\_append(\mathbf{st\_cons}(u,v),w) & = & \mathbf{st\_cons}(u,st\_append(v,w)), \end{array} \quad (4.6)$$

Let m and n be from $\mathcal{FAB}$, let $\angle$ be the above mentioned well-founded relation induced by the constructors of $\mathcal{FAB}$ and let us consider the following system of equations

$$\begin{array}{rcll} segm(m,n) & = & \mathbf{st\_nil}, & \text{if } n \angle m \\ segm(m,n) & = & \mathbf{st\_cons}(m,segm(\mathbf{just\_below}(m),n), & \text{if not}(\, n \angle m) \end{array} \quad (4.7)$$

If one compares (4.2) (or (4.3)) and (4.7), one can recognize at the first sight that (4.7) is not a system of equations that falls under (manipulable) standard or usual recursion of $\mathcal{FAB}$. However, proving that segm is a terminating recursive plan can be done very simply by formulating this termination problem as a *Recursion Manipulation* task (see [4]). This illustrates that $\mathcal{RM}$ represents a framework in which a logical justification of 'trouble-making', 'unusual' recursions can be verified user-independently.

If one compares the system of equations (4.7) and the lisp program for the function segment defined by Kunen and presented in section 2 of this paper, one can realize immediately, that, up to the domain description of the variables m and n, identifying cons and **st_cons**, identifying segment and segm, and abstracting from the presence of a hint in Kunen's definition, there is no difference. Yet, semantically, segm deals not with natural numbers.

4.3 *Recursion Manipulation* and two types of efficiency of recursive programs

[4] shows that non-usual recursion is, *usually*, making the theorem proving process much more complex, and thus, in a particular sense, inefficient. In consequence, two types of efficiency have to be considered. On one hand we have the usually considered *computational* efficiency of a program. On the other hand, a program becomes a 'manipulated object' in an $\mathcal{RM}$ system, for instance, when its termination has to be

proved or when it is relied upon in a formal specification of a program that has to be constructed (e.g., when a robot has to construct a program that tells him how to paint a segment).

Let us consider the function segm defined in (4.7) by non-usual recursion. For a robot, it is very natural to *work* with this particular program, the computational efficiency of this program is very reasonable. However, as soon as the robot has to 'think' in terms of this program (to find out, for instance, how to paint the segment whose limits are blocks m and n, i.e., the robot has to program itself to execute this task), the non-usual recursion of this program slows down the process of coming to such a program.

Thus, while [4] shows that the $\mathcal{RM}$ framework is able to handle the termination of non-usual recursion, in order to improve the internal efficiencyof a robot, (i.e., the rapidity of its 'thinking') it may be interesting to reformulatethe non-usual recursion definitions into their logically equivalent usual recursion forms. The framework of $\mathcal{CMM}$ allows this type of reformulation [4]. This is a very important result having theoretical as well as practical value.

To further understand its practical value, let us consider another, logically equivalent implementation for the function segm. We shall call stand_segm this new implementation and we shall define it by the equations :

$$\begin{array}{lcll} \mathrm{stand\_segm(m,n)} & = & \mathrm{gener(n)}, & \mathrm{if\ clear(m)} \quad (4.8) \\ \mathrm{stand\_segm(m,n)} & = & \mathrm{st\_comp(gener(}\mathbf{hat}\mathrm{(m)),gener(n))}, & \mathrm{if\ not(clear(m))} \quad (4.9) \end{array}$$

and the application condition : { Poss(stand_segm(m,n)) ⇔ $(x \angle y)$ or $(x = y)$ }, where gener: $\mathcal{FAB} \to \mathcal{STACK}$ is defined by :

$$\begin{array}{lcll} \mathrm{gener(x)} & = & \mathbf{st\_cons}\mathrm{(x,}\mathbf{st\_nil}\mathrm{)},\ \mathrm{if\ clear(x)} & \quad (4.10) \\ \mathrm{gener(x)} & = & \mathrm{st\_append(gener(}\mathbf{hat}\mathrm{(x)),}\mathbf{st\_cons}\mathrm{(x,}\mathbf{st\_nil}\mathrm{))}, & \mathrm{if\ not(clear(x))} \end{array}$$

and

$$\begin{array}{rcll} \mathrm{st\_comp(w,}\mathbf{st\_nil}\mathrm{)} & = & \mathbf{st\_nil} & \quad (4.11) \\ \mathrm{st\_comp(w,}\mathbf{st\_cons}\mathrm{(u,v))} & = & \mathbf{st\_cons}\mathrm{(u,st\_comp(w,v))}, & \mathrm{if\ not(memb(u,w))} \\ \mathrm{st\_comp(w,}\mathbf{st\_cons}\mathrm{(u,v))} & = & \mathrm{st\_comp(w,v)}, & \mathrm{if\ memb(u,w)} \end{array}$$

and finally

$$\begin{array}{rll} \mathrm{memb(u,}\mathbf{st\_nil}\mathrm{)} & \mathrm{is\ false.} & \quad (4.12) \\ \mathrm{memb(w,}\mathbf{st\_cons}\mathrm{(u,v))}, & \mathrm{if}\ u = w & \\ \mathrm{memb(w,}\mathbf{st\_cons}\mathrm{(u,v))}, & \mathrm{if}\ u \neq v\ \&\ \mathrm{memb(w,v)} & \end{array}$$

A simple syntactic analysis allows us to conclude that stand_segm defined by (4.8) and (4.9) is defined in terms of functions and predicates defined recursively by the *standard* recursion. In consequence, proofs relying on this particular implementation of the function segm (since segm and stand_segm are logically equivalent) are *unproblematic*. However, even though these formulations are logically equivalent, a programmer would probably prefer the implementation by (4.7). Recall again that, by its analogy to Kunen's definition, this particular implementation, i.e., (4.7), is 'forbidden' by 'Walhter's programming style'.

We can thus summarize: any inductive theorem proving system relying on $\mathcal{CMM}$ gives a robot the possibility of 'working externally' with non-standard programs (i.e., execute the programs defined even by non-standard recursion), while, internally, (i.e., in the process of recursive plans construction and verification) the robot can switch to a standard definition found (in the framework of $\mathcal{CMM}$) from the non-standard definition. As far as programmers are concerned, $\mathcal{CMM}$ preserves the programmers' liberty to keep their own programming style. This, with respect to *mathematical tradition* is natural, but it is unusual with respect to today's standards which permit the suppression of this liberty explicitly (as in [10]) or implicitly (as in the current user-dependent approaches to the 'automation' of the verification and/or construction of programs). In consequence, it is up to the potential user of these two possible tendencies to choose the one that is correct and suitable for his own purposes.

Conclusion

It is known that constructing robots that execute some tasks in environments which cannot be fixed by a 'finite description' in advance, requires some kind of inductive definition of this environment [16], [9]. Such an inductive structure, in some well-specified cases, justifies induction principle as a way of proving the properties of this environment, and it justifies recursion as a definition formation rule by which some properties of this environment can be expressed in a 'finite' way. The task of constructing a recursive plan from its formal specification (i.e., from a specification of the essential property that holds in the environment once this plan is applied to its input arguments) can be expressed as the task of checking a particular property of this environment. Thus, *RM* finds one of its applications in the design of autonomous robots. To achieve the somewhat ambitious goal of user-independence, *RM* must provide a rigorous *manipulable* theory of computable recursive programs. *Constructive Matching Methodology*, as one particular approach to *RM*, uses the *Theory of CDs* as such a manipulable theory.

This paper intended to simply *inform* roboticians about the relation of *CMM* to the design of autonomous robots and *illustrate* through an example how the following *CMM*'s properties manifest themselves : (1) the unity of *logical justifications* and *practical manipulations*, (2) an adequate treatment of the problem related to a logically justified and practically executable *reformulation* of the *objects* manipulated (i.e., programs), as well as (3) an adequate treatment of the *problems* innate to handling recursion and the solution of which may seem non-evident by a first sight. *CMM* succeeded in making the notion of undecidability *irrelevant* for *RM*. [5] recalls that *CMM* also succeeded in making the notion of the frame problem *irrelevant* for *RM* when extended by the mw-plan theory [9].

This paper thus aimed at motivating roboticians to extend the design of future robots, while taking into account the *full potential* of *RM*. In particular, we wanted to illustrate that we do not ask roboticians to become experts in handling recursion, we just wanted to illustrate that some experts in robotics have to make an effort to understand the *epistemological* justification and some essential features of *RM* in order to (get motivated to) formalize, in a particular way, the 'worlds' in which recursive plans are meaningful. These worlds have their own semantics and thus they cannot be a topic of *RM*. Since the mw-plan theory [9] is convenient for capturing the dynamic character of these worlds, the domain experts have to focus on the formulation of particular laws that govern the particular portions of worlds they are interested in. In [9], the *put-on-table* axiom (independent from the mw-plan theory) concerns a particular world, formulated by Manna and Waldinger, and well illustrates the semantic dependence of such particular laws. This paper illustrated that, if some more 'elementary' laws are formalized following the guidelines of the *Theory of CDs*, the use of 'recursive mode of thought' is automatically justified (by the *Theory of CDs*). As soon as such a formalization is performed, one can use *RM* to construct and verify recursive plans. On the other hand, since *recursion* (in its rigorous sense) is a particular kind of *representation* of a particular kind of *repetition*, a very particular kind of a 'short cut' of a very particular kind of 'potential infinity', we *feel* that the question of 'improving the *conception* of the visual perception of robots via a particular use of *Recursion Manipulation*' should not be neglected.

As we illustrated in this paper, it is a good feature of *RM* that the remaining experts in robotics do not necessarily have to stick to the resulting formalization, that the potential users of domestic robots do not need to know 'theories (theory of elementary recursive functions, first-order logic, situationistic logic, mw-plan theory), epistemological notions (theory, meta-theory, deductive system, axiomatic system, model, relativity of formal truth, ...) and the manipulable theories and meta-theories (*Theory of CDs*, *CMM*)' that are behind it. All these theories and notions *are* necessary for understanding the *architecture* and the *algorithms* of *CMM*. [4] explains under which conditions *RM* can be used for non-standard purposes, this time necessarily by interaction with the user. Namely, it can be used for (a) improving the computational efficiencyofprograms and (2) working out, a correct form of 'lines of code' that are given by the user and could thus be, in some sense, incorrect. The effortof one group of experts (in robotics) that has to perform a very rigorous work renders, via *RM*, the work for others (i.e., potential human users, if any) more informal.

Finally, this paper wanted to recall that the language of mathematics remains the only way to clear up ambiguities inherent to natural language, the only way to specify and control the real achievements of a deductive science. *Constructive Matching Methodology* (as an approach to *RM*) is built as a such science; this also means that the use of a particular form of inductive reasoning involved is, in some well-specified sense, rendered justifiable even in this deductive environment.

Acknowledgments

Many people have been consciously or unconsciously (thus voluntary or involuntary) helping the development of *CMM*, thus, implicitly, taking a part in the *positive* achievements of this paper. The limited space here forces us to rely on the acknowledgments given in [3]. Nevertheless, we take an opportunity to thank again Yves Kodratoff, who helped MF to think up the architecture of *CMM*. In the past four years we have highly appreciated the moral support of Laurence Puel, the former director of L.R.I. Thanks also to Christiane Weinzaepfel for her help in refining the informal conceptual presentation of *CMM* and to Yehuda Rav for helping to refine the formal conceptual presentation of *CMM*. Thanks to Lise Fontaine for helping to steady the mathematical content and the English of this paper.

References

[1] M. Franova: CM-strategy: A Methodology for Inductive Theorem Proving or Constructive Well-Generalized Proofs; in: A. K. Joshi, (ed): Proc. 9th IJCAI; 1985, 1214-1220.

[2] M. Franova: Constructive Matching methodology and automatic plan-construction revisited; Res. Report No.874, L.R.I., Orsay, France, November, 1993.

[3] M. Franova: A Theory of Constructible Domains - a formalization of inductively defined systems of objects for a user-independent automation of inductive theorem proving, Part I; Res. Report No.970, L.R.I., Orsay, France, Mai, 1995.

[4] M. Franova: Definitions Reformulation and Formulation: Why and How? (Saying 'Good-bye' to the notion of the well-founded relation); avail. as a report.

[5] M. Franova, M. Kooli: Recursion Manipulation for Robotics: Why and How?; to appear in proc. of EMCSR'98.

[6] J. Giesel: Termination Analysis for Functional Programs using Term Orderings; Proc. of SAS'95, LNCS 983, 1995, 154-171.

[7] G. Huet: Induction Principles formalized in the Calculus of Constructions; Programming of Future Generation Computers, Elsevier, 1988, 205-216.

[8] K. Kunen: A Ramsey Theorem in Boyer-Moore Logic; Journal of Automated Reasoning, 1995, Vol.15, No.2, 217-235.

[9] Z. Manna, R. Waldinger: How to Clear a Block: A Theory of Plans; Journal of Automated Reasoning 3, 1987, 343-377.

[10] D. McAllester, K. Arkoudas: Walther recursion; in: LNAI 1104, 1996, 643-657.

[11] C. Paulin-Mohring: Inductive Definitions in the system Coq Rules and Properties; in: LNCS 664, Springer-Verlag, 1993, 328-345.

[12] L. C. Paulson: Set Theory for Verification; J. of Aut. Reasoning Vol.15, No.2, 1995, 167-215.

[13] L. C. Paulson: A fixedpoint approach to implementing (co)inductive definitions; in: LNAI 814, Springer-Verlag, 1994, 148-161.

[14] A. M. Pitts: A co-induction principle for recursively defined domains; Theoretical Computer Science, vol. 124, No. 2, 1994, 195-219.

[15] R. Reiter: Proving Properties of States in the Situaton Calculus; Artificial Intelligence 64, 1993, 337-351.

[16] O. Stepankova, I.M. Havel: A Logical Theory of Robot Problem Solving; Artificial Intelligence 7, 1976, 129-161.

[18] P. Taylor: Intuitionistic Sets and ordinals; Journal of Symb. Logic, Vol. 61, No. 3, Sept, 1996, 705-744.

[19] C. Walther: On Proving the Termination of Algorithms by Machine; Artificial Intelligence, vol. 71, no. 1, 1994, 101-159.

Intelligent Systems Must Be Able to Make Programs Automatically for Assuring the Practicality

Takumi Aida and Setsuo Ohsuga

Department of Information and Computer Science
Waseda University

Abstract. This paper discusses an important issue on the intelligent systems to solve large scale problem, especially on the way of making computer programs automatically. Since an activity that human beings design a problem solving system is a kind of problem solving, a new modeling scheme that can deal with multi-level structures is necessary for representing problem solving itself. A large scale system used repeatedly should be a procedural program and not exploratory one. A method of translating modeling results into a procedural program is proposed in this paper.

1 Introduction

Computer systems are required to solve various problems. It is considered for an intelligent computer system necessary to satisfy at least the following requirements; autonomy, practicality and generality[1].

The issue of difficulty in developing large scale softwares is raised. This paper suggests the method of making problem solving systems automatically and of converting results into programs, including a new modeling scheme for representing multi-level concepts[2][3].

2 Problem solving method following human way

There are many types of problems such as analysis, design, control and decision making. Human beings solve these problems with a general method known as the scientific method today. It is represented formally and realized in a computer system. This section provides an outline description of the method.

2.1 Exploratory problem solving

Problems can be solved with such steps as follows; (1)to represent the problem explicitly, (2)to analyze and evaluate whether the representations satisfy the given requirements, and (3)if not, to modify the representations. This process is repeated until the goal is reached[4].

In order to apply this method to computer systems, the each operation in the steps require the specific knowledge base and a special knowledge base is used for controlling operations or guiding the exploration.(see Fig.1).

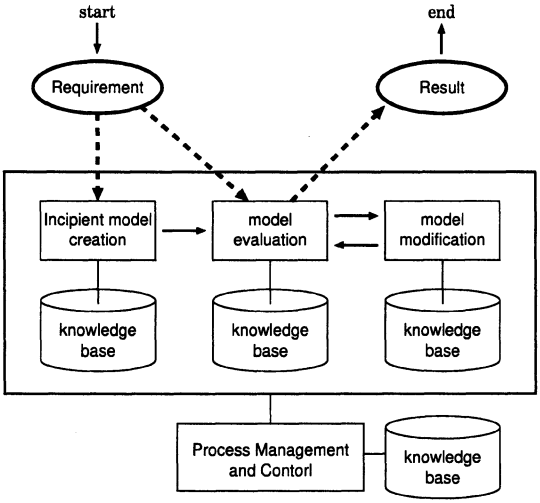


Fig. 1. Exploratory problem solving

2.2 Problem model

Problem model plays an important role in this scheme. At the incipient stage of this process, a problem is represented in a proper model by externalizing person's idea.

A modeling scheme to represent any object in the world must be defined. It is desirable that a single scheme can generate different type of object models [5]. Models must include such information as structure, functionality and relation. The structural information composes hierarchical structures or graph structures. There are the relations between not only the nodes of the structures but also the functionalities of the nodes.

In order to generate the system for automatic programming, human activity must be included in models, i.e., the modeling scheme must be able to represent a subject making an object do something. A computer program is a kind of automatic problem solving systems in itself. Therefore a programmer plays a role as a problem solving system that is required to design problem solving systems. This means that the model representation needs nested structures. Let it be called a multi-strata model (see Fig.2).

3 Multi-Layer Logic (MLL)

For processing the multi-strata model, a representation form to deal with multilevel knowledge architecture is necessary. A language to meet this condition is discussed in [6],[7]. This language is an extension of first order logic. The main

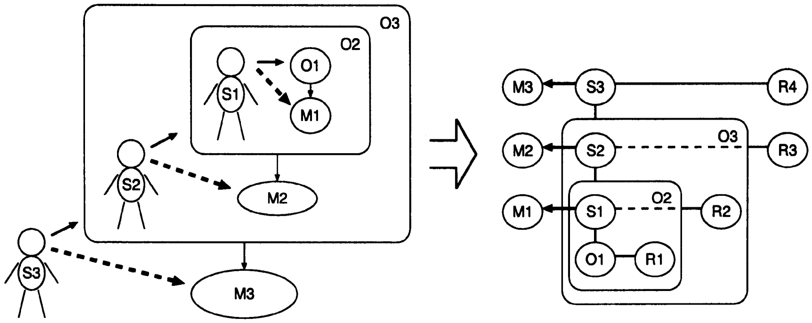


Fig. 2. Multi-strata model including human activity

extensions are (1) introduction of a method of manipulating data structures, (2) high order predicate under certain restrictions, and (3) their combination. The data structure is based on the axiomatic (Z-F) set theory and the language has a capability to define a new data structure as a compound of the primitive relations by a set of composing operations. The primitive relations are; is-a, component-of, product-of, power-of, pair-of. The basic form for representing knowledge is,

$$(Q_x x/X)(Q_y y/Y)\dots(Q_z z/Z)[R(x,y,\dots,z) : - \\ P_1(x,y,\dots,z) * P_2(x,y,\dots,z) * P_n(x,y,\dots,z)]$$

in which Q_x etc. denotes either a universal quantifier ($\forall$) or an existential quantifier ($\exists$), x/X etc. means x in a set X, $*$ denotes a logical connector, i.e., either conjunction ($\wedge$)or disjunction ($\vee$). This expression is read as "For all/some x in X, all/some y in Y, ..., all/some z in Z, if the relation P_1 and/or P_2 and/or ... P_n hold, then R". In the following however, the prefix is often abbreviated for the sake of simplicity of expressions.

A predicate is a mapping from a set of variables $(x,y,\dots,z)$ to either True or False (T,F). This mapping can be made by a procedure. A predicate to which this procedure is attached for evaluation is named a Procedural Type Predicate (PTP). The others are Normal Type Predicate (NTP). The system is designed to accept any procedure with its own PTP. When a predicate including a PTP is to be evaluated the procedure corresponding to this PTP is evoked and executed.

Any variable (say x) can be a structural variable to represent a structure. In this case X is a set of structures. .For example, if X is a product set of the sets $D_1, D_2, \dots, D_m$, then x represents a tuple $(d_1, d_2, \dots, d_m)$ of which $d_i \in D_i$. Moreover any variable can be a closed sentence which means a predicate without free variable. For example a predicate of the form $P(x,\dots,y,S(u,\dots,w))$ can be used. Any of the variables in S, i.e. $u,\dots,w$, must not be the same with the other variables in the predicate P, i.e. $x,\dots,y$.

As a combination of these two expansions, any variable can be a structural variable of which the domain is a structure of predicates. Using MLL as a knowledge representation language a knowledge based system named KAUS (Knowledge Acquisition and Utilization System) [8] has been developed by the author's group and used for many problems.

4 Meta-Operation for Multi-Strata Problem Modeling

Each problem requires its own problem solving method represented by a problem oriented structure of knowledge functions. In this structure, the form of multi-strata model must be included.

In order to create this model, general control knowledge apart from the problem-specific knowledge must be defined. The following (from *R*1 to *R*7) are some examples of predicates that makes a problem requirements in the form of 2-strata model, which is composed of "*Subject2*" and "*Subject*1", and that assigns *Subject*2 and *Subject*1 to the role of automating the activity *Subject*1 and of solving the problem respectively.

All predicates should be expressed in disjunctive normal form. This reason is taken up in the following section. A predicate beginning with "$" is PTP.

4.1 2-strata model

$$\begin{aligned}
&R1 : \mathit{automateActivity}(\mathit{Subject}2, \mathit{Subject}1, \mathit{System}) :- \\
&\qquad \$\mathit{getSubjectRequirement}(\mathit{Subject}1, \mathit{Requirement}1), && (1)\\
&\qquad \mathit{generateSystem}(\mathit{Requirement}1, \mathit{System}). && (2)
\end{aligned}$$

The terms *System* and *Requirement*1 denote a system to be generated and the requirement given to *Subject*1 respectively. This rule says that a system which replaces the lower stratum subject is obtained by, knowing the requirement given to it(1), and generating a system to satisfy the requirement(2). This operation(2) is defined below.

$$\begin{aligned}
&R2 : \mathit{generateSystem}(\mathit{exploration}(\mathit{Subject}1, \\
&\qquad\qquad\qquad \mathit{Model}, \mathit{Domain}), \mathit{System}) :- \\
&\qquad \$\mathit{getObjectRequirement}(\mathit{Model}, \mathit{Requirement}'), && (1)\\
&\qquad \mathit{problemType}(\mathit{Requirement}', \mathit{Type}), && (2)\\
&\qquad \$\mathit{makeRetrieveKey}(\mathit{Subject}1, \\
&\qquad\qquad\qquad \mathit{Model}, \mathit{Domain}, \mathit{Type}, \mathit{Key}), && (3)\\
&\qquad \mathit{retrieveKnowledge}(\mathit{Subject}1, \mathit{Domain}, \mathit{Key}, \mathit{KC}), && (4)\\
&\qquad \mathit{makeSystem}(\mathit{System}, \mathit{KC}). && (5)
\end{aligned}$$

There are the different rules for *generateSystem* by the requirements included therein. This is the one in which the requirement to *Subject*1 is to solve a problem. KC is an abbreviation of knowledge chunk. The type of problem must be identified from the representation of the requirement on *Model.* First the object level requirement is obtained(1). The problem type is obtained by the predicate(2). Then the necessary knowledge chunks are retrieved by (3)(4). Different rules are retrieved for the different problem types and domains. A problem specific problem solving system can be generated including these knowledge chunks(5).

$$\begin{aligned}
&R3 : retrieveKnowledge(Subject2, Domain, Key, KC) :- \\
&\quad \$getTaskKnowledge(Key, TaskKnowledge), && (1)\\
&\quad \$getDomainKnowledge(Domain, DomainKnowledge), && (2)\\
&\quad \$includeKnowledge(KC, \\
&\qquad\qquad TaskKnowledge, DomainKnowledge). && (3)
\end{aligned}$$

The first predicate (1) takes out the knowledge to define a specified task from the global knowledge base. The information on the knowledge chunk is in *Key*. Since the scheme of performing tasks is common to many domains, this task knowledge includes knowledge chunks that relate to different domains. The second predicate (2) selects as well the domain knowledge chunk. Finally, these knowledge chunks are integrated to form the knowledge chunks as required from the given domain.

4.2 Generating System

$$\begin{aligned}
&R4 : exploration(Subject, Model, Domain, K) :- \\
&\quad (incipientModel(Subject, Model, Domain), && (1)\\
&\quad repeatOperation(solveProblem(Subject, \\
&\qquad\qquad Model, Domain), K), && (2)\\
&\quad decomposeActivity(Subject, Model, Domain)). && (3)\\
&\quad \mid (\neg repeatOperation(solveProblem(Subject, \\
&\qquad\qquad Model, Domain), K), && (4)\\
&\quad makeAnswer(``non - solvable")). && (5)
\end{aligned}$$

This is a generalized rule to start problem solving. The predicate(2) is to repeat the operation inside the parenthesis, i.e., the actual, state-based problem solving, by the specified number. K denotes the number of repetitive operation (R6). *decomposeActivity* is added(3). When no solution could be found in the specified number of repetition K(4), it is reported(5).

$$\begin{aligned}
&R5 : solveProblem(Subject, Model, State, Domain) :- \\
&\quad (analizeModel(Subject, Model, State, Domain), && (1)\\
&\quad \$satisfyModel(Model, State)). && (2)\\
&\quad \mid (\neg satisfyModel(Model, State), && (3)\\
&\quad modifyModel(Subject, Model, State, Domain, Model'), && (4)\\
&\quad solveProblem(Subject, Model', State, Domain)). && (5)
\end{aligned}$$

First, the model is analyzed(1) and examined whether the model can satisfy the requirement(2). If not(3), the model is modified(4) and the process is repeated(5).

$$\begin{aligned}
&R6 : \mathit{repeatOperation}(\mathit{solveProblem}(\mathit{Subject}, \mathit{Model},\\
&\qquad \mathit{State}, \mathit{Domain}), K) :-\\
&\quad (\forall x/\mathit{integer}[1, K]) && (1)\\
&\quad [\mathit{solveProblem}(\mathit{Subject}, \mathit{Model}, \mathit{State}, \mathit{Domain}), && (2)\\
&\quad \$\mathit{countAndTest}(x)]. && (3)
\end{aligned}$$

This rule controls the repetitive operations. The repetition number is checked by (3) based on the variable defined in (1).

$$\begin{aligned}
&R7 : \mathit{analyzeModel}(\mathit{Subject},\\
&\qquad y(\mathit{Model}, \mathit{Value}), \mathit{Domain}, \mathit{System}) :-\\
&\quad \$\mathit{create2StrataModel}(\mathit{Subject2}, 1, \mathit{Subject}, \mathit{Model}), && (1)\\
&\quad \$\mathit{defineRequirement}(\\
&\qquad \mathit{automaticActivity}(\mathit{Subject2}, \mathit{Subject}, \mathit{System})) && (2)\\
&\quad \$\mathit{defineRequirement}(\\
&\qquad \mathit{analyzeModel}(\mathit{Subject}, y(\mathit{Model}, \mathit{Value}), \mathit{Domain})) && (3)\\
&\quad \mathit{evokeSubject}(\mathit{Subject}, *) && (4)
\end{aligned}$$

The predicate represents a general analysis operation including the selection of a method among many possibilities. It makes a structure of a 2-strata model by creating a new highest stratum subject, *Subject2*, and gives it a requirement *automateActivity* (2). Moreover the rule gives the lower stratum subject the requirement *analyzeModel* to obtain a specific analysis function to be executed using the selected knowledge chunk (3). Then the *Subject2* is evoked (4). It executes the requirement *automateActivity* again but, since the requirement included in the *automateActivity* is different from the one used before, the different rule for *generateSystem* is needed.

5 Automatic Programming for Assuring Practicality

A problem-specific problem solving system must be more practical than a simple system including all knowledge in a flat structure. An exploratory problem solving system is powerful but is not considered efficient enough. A procedural program is more practical if a problem can be solved algorithmically and the operation is repeated many times.

5.1 Problem which can be represented in the procedural form

Problems can be classified into programmable class and the others. A characteristic of a problem decides whether the problem can be represented in the procedural form or not. For the programmable class the models do not change during problem solving and the program makes a solution to an input in the deterministic way. The other problems need an exploratory problem solving. The task of programming is an exploratory problem in itself. Even if an object-level problem needs exploratory operation, its programming is still possible if the scope of searching solution is limited to the finite set. But a program needs the operation of selecting one from the set of alternatives, i.e. an inference operation. A program including run-time inference is usually inefficient. Compiling knowledge in advance and embedding its results into the program can make such programs faster. This type of problem can be the object of the automatic programming.

In other cases for dealing with the infinite alternatives, to represent problem solving in the form of procedural program is difficult because it implies unknown knowledge in the execution of the program. In this cases it is better to use a declarative form of expression with an inference engine.

5.2 The representation of the task of programming with multi-strata modeling

If a problem belongs to the programmable class, the multi-strata modeling makes automatic programming possible. The highest- and the second-stratum subjects of the model are given the requirements '*automate activity*' and '*make program*' respectively. The lowest stratum subject has already been given the problem-specific requirement. The highest stratum subject generates a system for making program for this requirement. A higher stratum subject conducts its processes by the first and the second step; on the first step the subject generates the representation of the program structure by solving problem and on the second step the subject convert the solution structure into a program code.

The first step can be divided into three operations in detail. (1) to select one (or more) instance input. (2) to find its solution. (3) to generalize the solution. The first step above is the problem dependent, i.e. requires the problem-specific knowledge chunks. On the other hand, the knowledge chunks using on the second step is common to all cases.

5.3 Generating a program code

An automatic programming system must be able to generate very complex programs. First of all components of a program structure is clarified, and the way to represent the each component in knowledge is defined in the following.

Structural components of program A complex program is composed of simpler programs. the simpler programs are constructed from a limited choice of structural components such as sequence, branch and loop. Thus the system must be able to represent these structural components and to generate any complex program structure by means of these components; (1) subroutine call, (2) branch,

(3) loop, and (4) database access. Because of the rack of the space, the detail of the explanation is abbreviated.

Knowledge representation suited for programming The exploratory problem solving system can find a sequence of predicates to reach the goal from the given problem. In order to generate a procedural program from the sequence, a special way of knowledge representation is necessary.

Some basic operations like branch are represented by two or more separate rules. Integrating these separate rules into a single form can simplify to translate the basic operations into procedural form.

Generation of a sequence of predicates as a source of programming In order to generate a sequence of predicates for automatic programming, the problem is solved by exploration and a deduction tree composed of the related predicates in the knowledge is produced, then tracing the problem solving can extract the succeeding path from the tree.

A program is required to be able to accept every instance in the scope of the input. One way to assure this is to try to solve this problem including the variables. This approach is not always successful because different instances may require different processing.

The other way is more effective. That is to generate an instance problem by specifying the constant value for the variable and solve the instance problem. The deduction tree produced by solving the instance problem is similar to or at the least a subtree of the tree for the general problem. If the form of given knowledge is not *disjunctive normal form*, the deduction tree of the instance problem can not contain all conditions and not be applied to generalizing the instance tree (see Fig.3 and [9][10]).

The deduction tree is generalized with studying the difference of the domains of the variables between the instance problem and the general problem and with matching the unifiability conditions for the predicates in the tree. For ease of explanation the case of single variable is shown here. Let the original query, the rule and the deduction be represented $(Q_s x/S)F(x)$, $(Q_t x/T)[F(x) : -G(x)]$ and $(Q_r x/R)G(x)$ respectively in which (Q_s, S), (Q_t, T) and (Q_r, R) are (the quantifiers, the domains of the variable x) of the query predicate, the rule and the conclusion respectively . The unifiability conditions are displayed in Table 1.

Transformation into a procedural program After the general tree is build, it is transformed into the procedural code. Every predicate at the node in the tree is processed one by one from the top node as follows;

(1) For every variable in a predicates, its type and scope are declared.
(2) Identify the predicate type such as procedural predicate / loop / branch / database access.

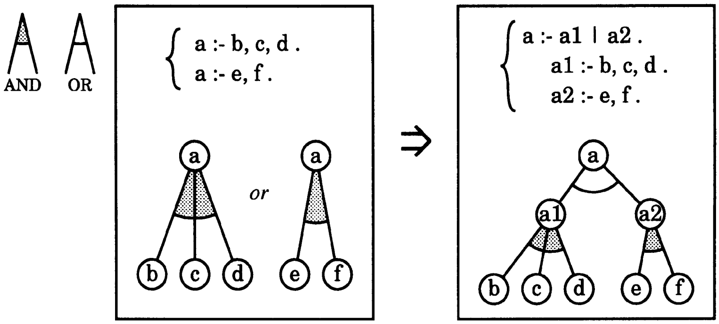


Fig. 3. Disjunctive Normal Form can cover all conditions in the deduction tree.

(Q_s,S)	(Q_t,T)	Unifiability Condition	(Q_r,R)
$(\forall,S)$	$(\forall,T)$	$S \subseteq T$	$(\forall,S)$
$(\exists,S)$	$(\forall,T)$	$S \cap T(=Z) \neq empty$	$(\exists,Z)$
$(\exists,S)$	$(\exists,T)$	$S \supseteq T$	$(\forall,T)$
$(\forall,S)$	$(\exists,T)$	None	(-,-)

Table 1. Inference rule of MLL

(3) Replace the procedural predicates with their accompanying procedure to generate the procedural program.

(4) For a loop type predicate, the loop type is analyzed and introduced for a specific structure with the additional operation if necessary, and the subsequence generated from the subtree is embedded in the loop.

(5) For a branch type predicate, a branch command is prepared, and the subsequences generated from the subtrees are combined with the branch.

(6) For database access, a generated access sequence is replaced the predicate.

6 Conclusion

This study has discussed the way to produce programs automatically. Moreover, it is instructive to consider that very large scale problems need decomposition into smaller problems[11]. The problems can be classified into two classes; programmable and non-programmable. The problems in the first class can be solved by computer and be translated into programs. The method presented in this paper can enlarge region of problems solving task to be assigned to computers, and reduce the task of people.

The experiments of this translation are carried out and still continuing. For further experiments, Genetic Algorithm, as one of exploratory problem solving methods, will be the aim of the experiments. The way to evaluate/modify models of Genetic Algorithm is algorithmic and different from those of rule-based exploration. If these can be represented by rules, a problem solving system with Genetic Algorithm is produced automatically.

It is expected that this method can be used to generate large scale problems and resolve software crisis with future[12].

References

1. S. Ohsuga, *Toward Truly Intelligent Information Systems,* (to appear in knowledge based systems), 1998
2. J. McDermott, *Preliminary Steps toward a Taxonomy of Problem Solving Methods, Automating Knowledge Acquisition for Expert Systems,* (S. Marcus ed.)Kluwer Academic, 225-256, 1988
3. F. H. Ross, *Artificial Intelligence — What Works and What Doesn't?*, AI Magazine, Vol.18, No.2, 1997
4. S. J. Lee and D. A. Plaisted, *Problem solving by searching for models with a theorem prover,* Artificial Intelligence, Volume 69, pp205-233, 1994
5. H. Yamauchi and S. Ohsuga, *Modeling Objects by Extensions and Intentions*, Information Modeling and Knowledge Bases III, IOS Press, 1992
6. S. Ohsuga and H. Yamauchi, *Multi-Layer Logic - A Predicate Logic Including Data Structure As Knowledge Representation Language*, New Generation Computing, 1985
7. S. Ohsuga, *A Way of Designing Knowledge Based Systems*, Knowledge Based Systems, Vol.6, No.1, 1995
8. H. Yamauchi, *KAUS User's Manual Version 6.49*, RCAST, University of Tokyo, 1997
9. Geoffrey G. Towell, Jude W. Shavlik, *Knowledge-Based Artificial Neural Networks,* Artificial Intelligence, Volume 70, Issue 1-2, pp119-165, 1994
10. H. Eriksson, Y. Shahar, W.Tu, A.R.Puerta, M.A.Musen, *Task modeling with reusable problem-solving methods*, Artificial Intelligence 79, pp293-326, 1995
11. W. Ikeda, *New Modeling Method for Using Large Knowledge Bases*, (to appear in proceeding of IEA-98-AIE)
12. W. Wayt Gibbs, *Software Chronic Crisis*, Scientific American, Sept., 1994

Towards a Knowledge-Level Model for Concurrent Design

Robin Barker, Anthony Meehan, Ian Tranter

Sheffield Hallam University, Pond Street, Sheffield S1 1WB, UK.
{R.Barker, A.Meehan, I.Tranter}@shu.ac.uk

Abstract. This paper describes the development and validation of a knowledge-level model of concurrent design. Concurrent design is characterised by the extent to which multidisciplinary perspectives influence all stages of the product design process. This design philosophy is being increasingly used in industry to reduce costs and improve product quality. We propose an essentially rational model for concurrent design using *Common*KADS and report on our validation of the model through studies with designers.

1 Introduction

A number of models of design exist, generally falling into one of two paradigms; the *rational design approach* [1], [2] or the *reflective practitioner approach* [3]. Models within each of these design paradigms rely, to a greater or lesser extent, upon studies of designers in practice.

Concurrent design is an approach to design which seeks to incorporate a number of different life-cycle perspectives relevant to a product at the product design stage e.g. manufacture, assembly, recylclability [4]. Concurrent design is distinguished from collaborative and simultaneous design by the high degree of multidisciplinarity that exists in the design team [5], [6]. It is possible to formulate models for concurrent design within either of the paradigms above. However, as far as we have been able to establish, the models and support tools that currently exist for concurrent design are firmly situated within the former paradigm.

This paper proposes a 'rational' model for concurrent design and describes our attempt to validate it on the basis of studies with several designers. We have studied designers at work and have abstracted protocol and interview transcripts, re-representing them in the style of concept graphs in a way which allows us to approach the formalism of a KADS expertise model [7]. The initial models, obtained from transcripts, use the vocabulary of the designers in order to allow us to discuss the models with them in follow up interviews. We then abstract from the descriptions of the individual designers to develop a model which we compare with our originally proposed model.

2 Models Of Design

A number of models have been developed to enable machines to contribute to the design of artefacts. Kruger and Wielinga [8] suggest a design problem is specified in terms of functional requirements, non functional requirements and constraints. They suggest that solutions to most industrial design tasks are generated through a process of 'decomposition - solution - recomposition'. Chandrasekaran [9] views the design process as a series of sub tasks: propose - critique - modify. A number of models of the critiquing step have emerged [10], [11], [12] and approaches based upon multiple domain critics have been proposed [13], [14]. Klein and Lu [15] look at deadlocks that can occur when different experts offer conflicting critiques of a design. Werkman [16] explores negotiation as an aid to resolving such conflicts. Molina [17] reviews computer aids to the simultaneous engineering process.

Smithers [18] is critical of most knowledge-based systems which support design suggesting that, in the absence of any useable theories of the design process, they are characterised by what computer programs can be made to do, rather than what really goes on when professional designers design. He argues that what is required are knowledge-level theories of design which are developed with reference to lessons learned in the field of knowledge engineering. Research which seeks to achieve this is ongoing and is acknowledged to be at an early stage, both in terms of determining appropriate methodologies [20], [21], [22] and developing models which span the diversity of design practice [23], [24], [25], [26].

3 *Common*KADS Models For Design

*Common*KADS considers design to be a synthetic task. The original KADS model for design seems to correspond closely to design models presented in established design texts [1], [26]. KADS also includes two refinements of the design task, hierarchical and incremental design. The *Common*KADS models for design view the activity as having two distinct phases [27]. Analysis is seen as translating the 'needs and desires' of a customer into requirements; a form of requirements engineering. In practice, concurrent design relates mainly to the synthesis stage and so determines the focus of the work reported here.

3.1 Existing CommonKADS Design Task Models

Bernaras and Van de Welde [27] develop design models viewed from three different viewpoints. From the user requirements dimension they present models for routine, innovative and original design. From the types of static design knowledge available they outline models for case based design, transformational design, decomposition-based design and generic model. From the construction dimension they consider allocation, configuration and parametric design. Kingston's [28] model for exploratory design is analogous to the *Common*KADS model for original design -

i.e. the requirements for the design evolve in parallel with the design synthesis process.

Bernaras and Van de Welde [27] characterise a problem statement as abstract customer ‘wants’ (needs and desires) which are refined to criteria (requirements descriptions) and then further refined to constraints (problem statements). In the engineering field, it is believed that constraints are not merely formally expressed criteria, they are subtly different entities. The distinction between functions and constraints may be hard to pin down [9]. We share the view of a problem statement as containing functions (criteria which dictate what the design solution must comply with) and constraints (which dictate to an extent how the design solution is achieved).

Constraints may be customer dictated or emerge from life-cycle perspectives. These can be termed external and internal constraints respectively. (The latter may be compared with Lawson’s definition of internal constraints [29].) Internal life-cycle constraints are key to the concurrent approach in that they represent how downstream perspectives can affect the design.

3.2 Concurrency And Existing Models For Design

The *Common*KADS and other design models discussed so far do not explicitly make any reference to the concurrent consideration of conflicting constraints on the design process [8]. Maher considers design evaluation, but this is concerned with checking how a partial or fully specified design complies with the expected performance of the design. Kingston's model [28] addresses the issue of design constraints being evaluated, however this appears to be after a complete design has been generated.

The essence of concurrency is that diverse, often conflicting, constraints are evaluated continually as the design evolves. Analyses of designers working in industry suggests that designers do work in a concurrent manner Sonnenwald [25] although Kruger and Wielinga [8] would suggest that this impression may be deceptive. We have been unable to find any specific reference in the literature to knowledge-level models for concurrent design.

On the basis of existing models, together with discussions with academics, and analysis of the available design literature, we have developed an essentially rational model for concurrent design which we have subjected to validation.

4 A Knowledge Level Model For Concurrent Design

A task model for concurrent design must make explicit how internal downstream perspectives influence decisions during the design process. It is believed that an ‘idealised’ concurrent designer or design team work in the following way :-

- With knowledge of previous designs and the product specification, the designer proposes a solution (or a partial solution). This modifies the current design model in some way to give the 'advanced' design model.

- The proposal is then critiqued from a number of different concurrent perspectives. The output from critiquing consists of further internal life-cycle constraints which were not documented in the design specification, an indication of whether or not the proposal is acceptable, and the critique itself.

- A phase of negotiation then determines which additional functions and constraints will reformulate the design specification and a new series of design proposals are then made.

Concurrent design can therefore be seen as a cycle of subtasks; 'propose - critique - negotiate', with a revised, 'advanced', design resulting from each cycle. The model which emerged from our analysis is given in Figure 1. The negotiation process updates the current design problem with new constraints (derived from the critiquing process) and/or new functionality (which comes from the initiator of the design).

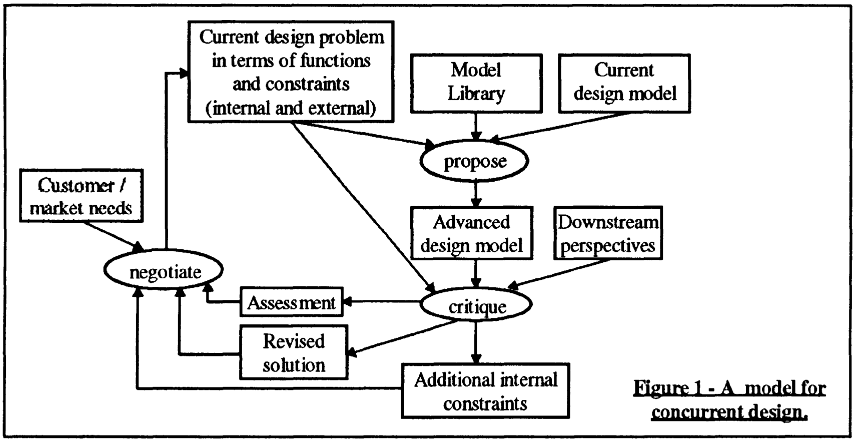


Figure 1 - A model for concurrent design.

Concurrent critiquing pinpoints additional internal constraints and so differs from the 'critiquing' defined by Chandrasekaran [9] and Goel [30] who view it as unearthing missing functionality in a completed design. In the next sections we describe how our model was subjected to validation by assessing the degree to which it accords with the practice of concurrent designers.

5 Methodology

Validation and refinement of the proposed model (Figure 1.) has proceeded by analysing design activity of three manufacturing companies and of two consultant industrial designers. All consider themselves to be operating in line with concurrent engineering principles. Although all are designing within the context of mechanical engineering, the nature of the products manufactured differ significantly. This enhances our confidence in the generality of the model we develop.

A number of techniques are available to capture and analyse an expert's problem solving behaviour. In particular, protocol analysis is seen as a valid approach (given awareness of some limitations) [20]. Our chosen method has been to both observe and to conduct focused follow up interviews with designers and other personnel involved in product design. The resulting protocol and interview transcripts have been used to generate graphical descriptive models. In early follow up interviews, the established KADS models for design and our proposed model for concurrent design were used to stimulate discussion and clarify designers understandings of the modelling process. Subsequently, the designers have discussed and refined the models constructed from their transcripts. These have then been abstracted towards models which use more established KADS terminology. The observations and discussions took place in the workplace. Designers were encouraged to work in as normal a manner whilst verbalising about information that they were considering.

In order to enable the designers to relate directly to the graphical representations of the models, the vocabulary of the transcripts is used to label the graphs. We found this important in facilitating comment on the developing model. As an example, the graph in Figure 2 was derived from a transcript of a recorded session with an industrial designer and was verified in subsequent discussion. It shows his model for the overall design process. Some of the subtasks represented are not simple inference steps. 'Inform', 'model' and 'produce' have their own inference structures and these were explored at follow up meetings with the same expert. Using the same example, a task structure for 'model' was further explored with the designer and this is illustrated in Figure 4.

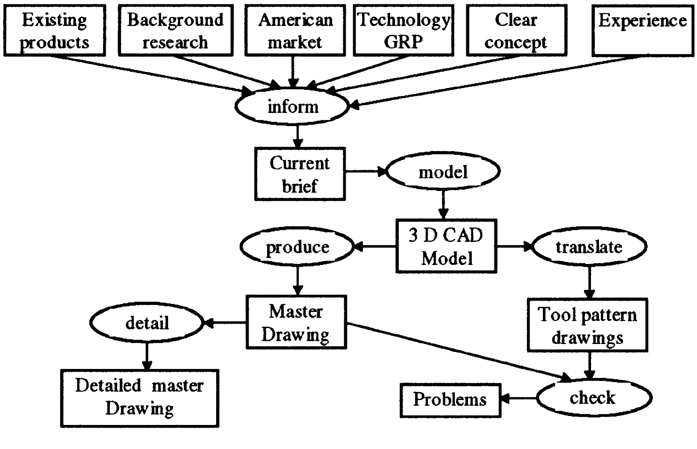


Figure 2 - An industrial designer's 'design task model'.

The designer uses the term ‘model’ in both Figures 2 and 4, however he considers them to be different, the model in Figure 2 being more all encompassing than that in Figure 4. The 'model' subtask identified in Figure 4 is iterative in nature. Clearly, at this stage we are still using the vocabulary of the designer to label the graphs.

6 Generic Models From The Case Studies

Having used the designers’ own language to verify the graphs obtained from the transcripts, we have then abstracted from these graphs towards a generic model, mapping the designers’ informal linguistic terms onto more established KADS terms.

6.1 A model for concurrent design

Again, by way of example, the subtasks ‘inform’, ‘model’ and ‘detail’ in Figure 2 have been mapped to the terms ‘analyse’, ‘develop’ and ‘refine’. Similarly, ‘Current Brief’, ‘3D CAD model’ and ‘Detailed Master Drawing’ correspond to the roles ‘formal specification’, ‘conceptual model’ and ‘detailed design. Not all mappings are as unproblematic. The input roles to the ‘inform’ step in Figure 2 do not simply map on to informal specification. Whilst ‘American Market’, ‘Background Research’ and ‘Existing Products’ can be related to a market driven ‘informal problem statement’, ‘Technology - GRP’ was found to represent both a life cycle constraint and an early commitment to a particular technical solution (the role ‘Clear concept’). At the stage of developing the formal specification, a technical solution is clearly already in the head of the designer.

Based on analysis of the transcripts of all of our designers, it is possible to abstract a generic model for the concurrent design process. This abstract model, shown in Figure 3, shares similarities with the KADS model for design but additional knowledge roles input to the different subtasks are identified. Concurrent constraints impinge on the design process in a three pronged manner.

At an early stage of the design process (specification development and early conceptual design) the designers seemed to use their own knowledge of these external constraints to guide them. The designers have a general working knowledge of manufacturing, assembly and other constraints and use this knowledge to guide them utilising some form of ‘internalised’ critiquing to evaluate a design. It seems to be in the later (synthesis) stages that the designer draws most heavily on the expertise of others in the team.

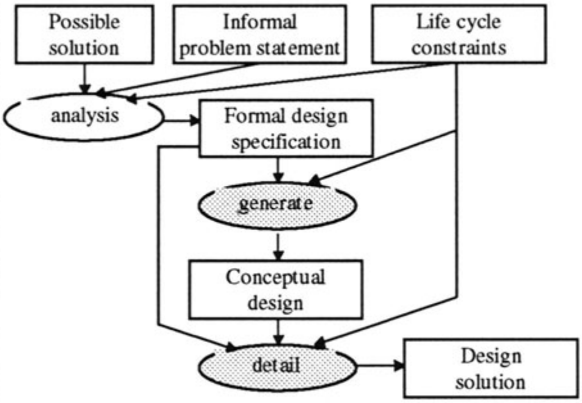


Figure 3 - A generic task structure for mechanical design.

Every time the current design solution is advanced critiquing of the evolving design occurs as a feature of both 'generate and 'detail'. The expansion of these two subtasks then follows a process of 'propose - critique - negotiate' and this is characteristic of concurrent design.

6.2 'Propose - Critique - Negotiate'

By focusing on the designers' understanding of 'propose - critique - negotiate' (Figure 4 represents the graph obtained from one such designer) we have attempted to arrive at a model for this process. For example, in Figure 4 one designer generates a number of 'Design answers' from the 'Current brief' and then outlines the problems he sees as inherent in the 'Design answers'. (The 'Pivots and attachments' role could be seen as part of the 'Design answers' role although the designer has highlighted these as separate roles because he perceives them as important). There is clearly a phase of 'propose - critique' on the designers part before the 'model' task is used to transform the given 'Design answers' into a '3D CAD Model'. There is then another phase of critiquing where the '3D CAD model' is critiqued from different life cycle perspectives ('Assess', 'Manufacture evaluation' etc.) and a number of conflicts and constraints ('Clashes in assemblies', Manufacture problems etc.) are outlined. Also, the 'invent' subtask, introduced to overcome the CAD modeller constraints, can be seen as a applying 'propose' methods to the design tools and techniques as well as the product of the design.

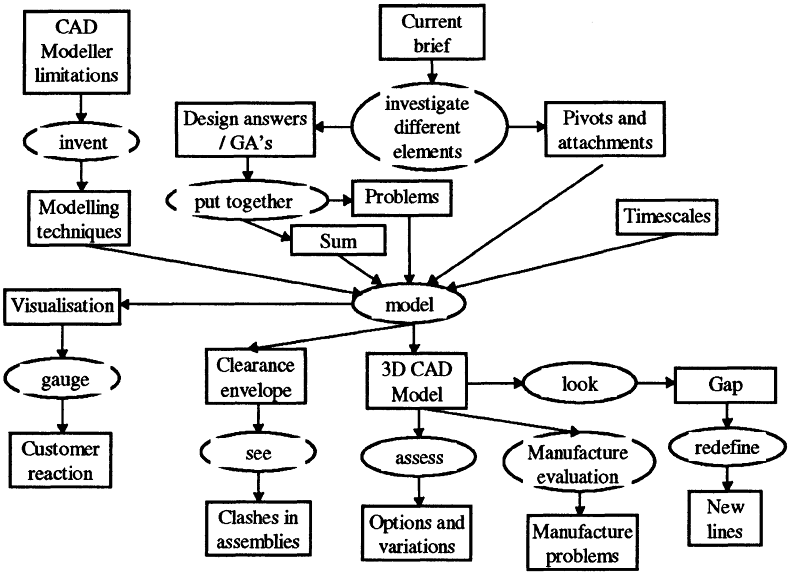


Figure 4 - Concurrency within the modelling sub task.

From transcript graphs such as those in Figure 4 and from transcripts of interviews with our designers, we have established that further concurrent constraints introduced by critiquing do not always imply some negotiation phase to update the brief but these extra constraints may simply act as input roles to a process of amending the current design model.

From the analysis illustrated above it is possible to formulate a more generic model to show how downstream concurrent constraints impinge on the design process in the expansion of 'generate' and 'detail' of Figure 3. This is shown in figure 5.

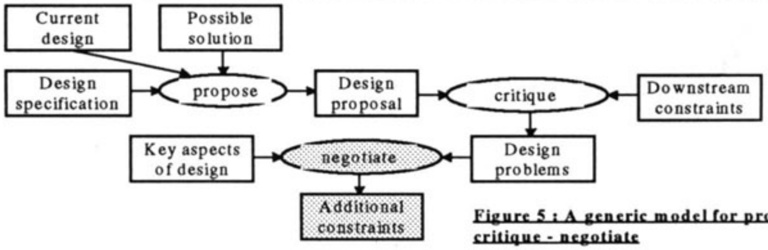


Figure 5 : A generic model for propose - critique - negotiate

The terminology used in Figure 5 is designed to encompass the terminology derived from the different case studies. i.e. 'propose' encompasses 'develop', 'generate' etc., 'critique' encompasses 'assess', 'evaluate' etc. Figure 5 shows that a design proposal is subject to critiquing from different perspectives to outline potential problems inherent in the design proposal from that perspective. These are expressed as constraints. There is then a complex process whereby the design proposal is altered on the basis of these 'design problems'. The case studies generally use a term such as 'alter' or 'modify' to cater for this process. However it is clear that there must be some form of negotiation to determine which constraints are allowed to influence the next 'propose' cycle. This negotiation process takes the 'key aspects of the design' as an important input role, resulting in constraints which are allowed to influence the next 'propose' process. However, how these additional constraints then influence the next propose step is not yet evident and open to debate among designers.

Inference diagrams alone do not readily convey some of the temporal features of design activity. For example, we found that internal life-cycle constraints were given different weight as a design evolved. One design consultant felt that a preoccupation with manufacturing and assembly constraints at an early stage could limit his freedom. However, he did like to know what these constraints were at an early stage so that any conceptual design produced was 'informed' by these constraints (i.e. the designer does not want to produce 'ridiculous' designs which are 'impossible' to manufacture and assemble). One of the companies in the study purposefully avoided considering downstream internal constraints at the conceptual design stage as it is felt that this limited the designers creativity.

7 Conclusion

Internal (life-cycle) constraints play an important part in the concurrent design process. Further it is believed that the design evolves via a process of 'propose - critique - negotiate' both at the stage where a (revised) conceptual design is arrived at and in the detailing of that conceptual design. This process is used to unearth and assess additional internal constraints during the design process. At the early stages of design, it is the designer themselves who applied this thinking on the basis of their knowledge and past experience. It is only later on that the designer draws upon the concurrent design team.

A refinement of the KADS model for design has been proposed which explicitly incorporates features characteristic of concurrent design. These are knowledge of downstream, life-cycle constraints and expansion of the 'generate' and 'detail' inference steps in such a way as to reflect the process of 'propose - critique - negotiate' that the design teams use. The model has been partially validated by analysing the activity of designers and consultants working in industry.

We believe the concurrent design model presented represents the design processes currently used by small to medium mechanical engineering enterprises.

8 Future Work

We wish to clarify the negotiating strategies designers and concurrent design teams utilise to determine how internal constraints are allowed to affect the design and to explore the way in which constraints are not 'one way' in that designers push against constraints e.g. by influencing the development of, or extension to, the capability of a manufacturing processes.

We also wish to analyse the differences in approach between the companies and designers we have worked with. Differences are believed to depend upon a range of human and sector specific factors.

References

1. Pahl, G., Beitz, W.: Engineering design. (Copyright the Design Council). Springer Verlag (1984)
2. Simon, H.: The Sciences of the Artificial. MIT Press (1969)
3. Schon, D.: The reflective practitioner: how professionals think. Avebury (1991)
4. Shiva Kumar, H., Suresh, S., Krishnamoorthy, C. S., Fenves, S. J., Rajeev, S.: GENCRIT : A tool for knowldege based critiquing in engineering design. Artificial intelligence for engineering design, analysis and manufacturing, **8**, Cambridge University Press (1994) 239-259
5. Desa, S., Schmitz, J.M.: The development and implementation of a comprehensive concurrent engineering method: theory and application. SAE Transactions 100 (1991) 1041-1049.
6. Schmidt, RF., Schmidt, M.: Computer aided concurrent integral design. Springer-Verlag (1996)
7. Breuker, J., Van de Velde, W. (Eds.): CommonKADS library for expertise modelling: reusable problem solving components. IOS Press (1994)
8. Kruger, C., Wielinga, B.: A KADS model for the industrial design task. Proceedings of the third KADS meeting, Munich, Germany (1993)
9. Chandrasekaran, B.: Design problem solving: A task analysis. AI magazine. Winter 1990
10. Silverman, B.G.: Survey Of Expert Critiquing Systems: Practical And Theoretical Frontiers, Communications of The ACM. **35:4** (1992)
11. Fischer, G., Mastaglio, T.A.: Conceptual Framework For Knowledge-based Critic Systems. Decision Support Systems **7:4**, (1991) 355-378
12. Fruchter, R., Clayton, M,J., Krawinkler, H., Kunz, J., Teicholz, P.: Interdisciplinary Communication of Design Critique In The Conceptual Design Stage, Computing In Civil And Building Engineering. Proc. Fifth Intl. Conf. On Computing In Civil And Building Engineering ICCCBE. ASCE, New York (1993) 377-384
13. Barker, R., Short, C., Tranter, I., Meehan, A., Parkinson, B.: Modelling the Concurrent Engineering Environment Using Artificial Intelligence Techniques. AIENG '95; 10th International Conference on Artificial Intelligence in Engineering, 4-6 July 1995, Udine, Italy.

14. Barker, R., Short, C., Tranter, I., Meehan, A., Parkinson, B. Critiquing and the concurrent design process. In: Meehat (Ed.) Proc. 3rd International Conference on Concurrent Engineering and Electronic Design Automation: CEEDA-96. Poole UK: EPIC (1996)
15. Klein, M., Lu, S.C.: Conflict resolution in cooperative design, Artificial intelligence in engineering, **4:4** Computational Mechanics Publications (1989)
16. Werkman, K.J.: Negotiation As An Aid In Concurrent Engineering. Issues In Design / Manufacture Integration **39** ASME (1991) 23-30
17. Molina, A., Alashaab, A.H., Ellis, T.I.A., Young, R.I.M., Bell, R.: Review of computer -aided simultaneous engineering systems. Research in engineering design: theory applications and concurrent engineering. **7:1**, (1995) 38-63
18. Smithers, T.: On knowledge level theories of design process. In: Gero, J.S., Sudweeks, F. (eds.) Artificial intelligence in design. Kluwer academic publishers (1996)
19. Davies. S.P.: Effects of concurrent verbalisation on design problem solving. Design Studies **16:1** (1996). 102-116
20. Dorst. K.: Analysing design activity: new directions in protocol analysis. Design Studies **16:2** (1995) 139-142
21. Lloyd. P., Lawson, B., Scott, P.: Can concurrent verbalisation reveal design cognition? Design Studies **16:2** (1995) 237-259
22. Lloyd, P., Scott, P., Discovering the design problem. Design Studies **15:2** (1994) 125-140
23. Kolodner. J.L.: Powers of observation in creative design. Design Studies **17:4** (1996) 385-416
24. Visser, W.: Two functions of analogical reasoning in design: a cognitive psycology approach. Design Studies **17:4** (1996).
25. Sonnenwald, D. H.: Communication roles that support collaboration during the design process. Design Studies **17:4** (1996) 277-301
26. Pugh, S.: Total Design: Integrated Methods For Successful Product Engineering. Addison Wesley (1991)
27. Bernaras, A., Van de Velde, W.: Design. In: Breuker, J., Van de Velde, W. (eds.) CommonKADS library for expertise modelling: reusable problem solving components. IOS Press (1995) Chapter 8
28. Kingston, J.K.C.: Design by exploration : A proposed CommonKADS inference structure. Artificial Intelligence Applications Institute, University of Edinburgh, AIAI-TR; 155 (1994)
29. Lawson, B.: How designers think. Butterworth (1990)
30. Goel, A.: Integration of Case-based reasoning and model-based reasoning for adaptive design problem solving, PhD dissertation, Dept. of Computer and Information Science, Ohio state University, US (1989)

Multiagent AI Implementation Techniques: A New Software Engineering Trend

Cyrus F. Nourani

ABSTRACT We present techniques for design and implementation of software systems with AI and concurrent S.E. techniques. The stages of conceptualization, design and implementation are defined by AI methods. Software systems are proposed to be designed through knowledge acquisition, specification, and multiagent implementations. Multi-agent implementations are proposed to facilitate a fault tolerant software design methodology, applying our recent research that has lead to fault tolerant AI systems. A particular approach to and an AI formulation of designing fault free and fault tolerant software is presented, which is based on the agent models of computation. Design with objects and a novel multi-kernel approach is presented. A system is defined by many communicating pairs of kernels, each defining a part of the system, as specified by object level knowledge acquisition. An overview to agent morphisms and algebras are presented and
applied to the design.

Keywords Concurrent Software Engineering with AI Agents, Multi Agent AI Techniques, Abstract Multi Agent AI Design, AII, Agent Morphisms

Project_METAAI@CompuServe.com

Correspondence P.O. Box 278, Cardiff By The Sea, CA. 92007, USA
USA Academic Address UCSB when at University

1. INTRODUCTION

Techniques are presented for design and implementation of fault tolerant software systems with new trends in artificial intelligence. Software fault tolerance is an area of crucial importance to which we claim AI techniques might be applied gradually to solve real problems encountered in fields such as intelligent systems, aerospace, robotics, etc. In fact one of the reasons that traditional software design methodologies(see [1], for examples) do not account for a fault tolerance is the lack of a coherent fault-tolerant AI/software methodology. It is difficult to design software that is fault tolerant without a degree of intelligence built in to the design techniques. Defining software fault-tolerance with AI techniques is new [12,10], while there is a well-defined discipline of fault tolerant computing in general(see for example [8,9,5]). Thus the AI approach presented in this paper could be considered a new trend. AI system designs go through the stages of Conceptualization , Design, and Implementation Each of the stages has to be approached in ways that minimize human error and enable the designed system to automatically recover from faults.

In the present paper software design is viewed as a methodology that commences with a knowledge acquisition phase, followed by a specification phase, and concluded by a system realization phase. The present approach defines knowledge acquisition for software fault tolerance, system specification for fault tolerant software (FTS), and system realization for fault tolerant software systems. Knowledge acquisition includes exception knowledge as an essential component, as does system specification. System realization is by independent concurrent computing agents.

The present paper defines FTS by a pair of systems, each consisting of many computing agents. The two systems are mutually synchronized to enable fault and exception handling and recovery in an automatic manner. The paper also presents AI techniques for implementing FTS. The proposed methods have been pointed out in the context of problems that concern human error and expert judgement in AI in a brief by this author [12].

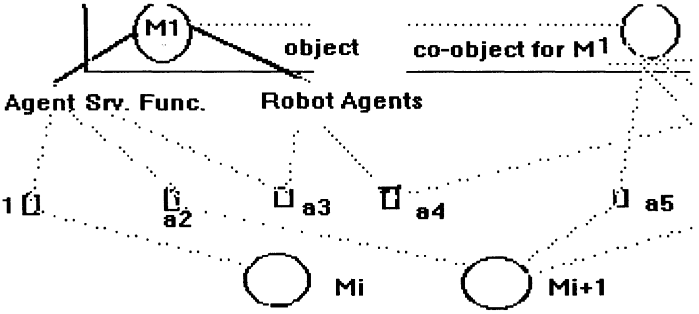
2. KNOWLEDGE ACQUISITION AND SPECIFICATION

The initial phase of designing fault tolerant software (FTS) is to present the design in form of a specification[2]. The approach here is to start with a knowledge acquisition phase. This requires either interviewing an expert, brain-storming with a group of experts, or structuring one's thoughts if the specifier is the expert. We present the notion of Fault Tolerant Knowledge Acquisition (FTKA) in this FTKA is formulated to deal with the conceptualization stage. This is a crucial stage to the design of a FTS. The techniques as they apply to the requirement specification problem are further developed by the author in [3]. It requires the user to inform the specifier as to the domains that are to be expected, i.e. what objects there are and what the intended actions (operations) on the objects are, while fully defining such actions and operations. The actions could be in form of processes in a system. The relations amongst the objects and the operations (actions) can be expressed by algebras and clauses, which the specifier has to present.

Thus specification are triples <O,A,R> consisting of objects, actions and relations. Actions are operations or processes. However, the problems of abstraction [1], object-level programming, and agent view of AI computation are the important components of inter-play in the present paper. A structural design technique with objects is defined to address knowledge abstraction[4] problems. FTKA has some additional requirements to be put forth here. The requirement is that each object to be defined has to have a dual definition in terms of the actions that are taken for exception and recovery. Thus at the knowledge acquisition phase the expert is to state all exceptions to actions and what recovery and corrective actions are to be carried out. Thus for each action on an object a dual action is to be supplied through FTKA, such that a specifier can fully define the effect of the dual actions. Thus the exception and fault actions are fully specified in each case. The reader might ask how such dual conceptualization can be conceived in design. The way design proceeds as viewed by the present approach, is to identify objects actions and relations defining functionality. When defining actions on objects the specifier must have a set of preconditions to check before an operation or action is permitted. Such set of preconditions then automatically form a set of processes to check for validity of an operations. This implies that a dual set of actions are to be defined on the objects that can take the failure of a precondition and define the alternate recovery actions or remedial actions that are to take form on the objects. This is illustrated by the example in section 4.

3. MULTI-AGENT REALIZATION OF SYSTEMS

The term "agent" has been recently (see [6], for example) applied to refer to AI constructs that enable computation on behalf of an AI activity. It also refers to computations that take place in an autonomous and continuous fashion, while considered a high-level activity, in the sense that its definition is software/hardware, thus implementation, independent [1]. Such functionality is typical of what is required to implement complex autonomous planning systems[2]. Agents are in most cases informable, thus allowing message passing actions. Thus cooperating expert systems can be viwed as simple examples for the agent models of AI systems. We can define fault tolerant software systems designed by AI methods as intelligent agent architectures [6], with external behavior that is a function of the degree of message passing actions and parallelism conceptualized. The fault tolerant specific architectural issues are dealt with further in the next two sections. Since our specifications consist of objects, actions, and relations defining the effect of actions on objects, it is not difficult to show that they can be implemented by a collection of active agents that communicate through their operations, parameters and messages as conceived by the FTKA phase. The specifications <O,A,R>, once expanded further, are in fact of the form <O,(A,F),(RNA,RFA)>, where A is actions, F is faults, and (RNA,RFA) their respective relations, NA for normal action and FA for fault action. In the example of the next section many objects and their fault functions are presented. Now we present a necessary twin-engine turbojet <FTN,FTF>, consisting of FTN := <O,A,RNA> and FTF:= <O,F,RFA>, for FTCS with AI techniques. Each of the FTN and FTF consists of agents that are mutually, often pair-wise, informable. <FTN,FTF> defines a pair of systems, each consisting of a collection of objects, actions and relations. Actions could be in form of operations or message communication from one object to another. In other words, a collection of computing agents forms FTN and a dual collection forms FTF. This defines a concurrent collection of systems that can be realized by agents that logically or physically be thought of as running on several microprocessors. The degree of fault tolerance is a function of the intelligence of the agents implementing the <FTN,FTF> pair. The agents have incomplete information about the immediate needs of activating other agents or exceptions. Thus the efficiency and fault tolerance of our software systems are a function of the degree of intelligence we build in the implementing agents. The agents must have some reasoning ability to make transition or message passing decisions. This approach allows us to design systems that can deal with unplanned or erroneous behavior in an AI system.



In the figure above the agents ai are implementing functions that are defined on the objects.The next step is defining the <FTN,FTF> from the fault tolerant knowledge acquisition (FTKA) inputs. Its realization consists of a pair of communicating systems to be defined in the following sections (see figure). This approach has a mathematical computing model, to be presented in subsequent future expositions, consisting of an algebra of processes and objects. Preliminary theoretical papers have been written by this author during the current year, supporting the computing techniques applied here.

4. AN AUTONOMOUS MULTI AGENT DESIGN PROBLEM

In this section we take the reader through the design stages of a typical autonomous system for real applications involving many modules of independent functionality. The method of design an implementation by multi agent AI techniques is then illustrated. A typical complex problem domain is the design of autonomous space flight system. The AI-Aerospace expert is presented with a stack of documents, often volumes of textual narrative description of a system from the point of view of an aerospace scientist or engineer. From these volumes the expert is to come up with a set of functions, each corresponding to a space flight module, and then to define how the modules interact and function together, by defining some operations amongst the modules. The modules are each complex hardware-software systems, best thought of as a microprocessor with its own running software, that implements the functionality of the module as extrapolated from the design process.

Let us view one sample set of modules: M1-M8, each representing a functionality Fi, respectively.

M1: Thrust Control; M2: Stage Control; M3: Orbit Selection; M4 : Attitude Control M5: Flight Deck Control; M6: Sensors; M7: Obstacle Avoidance; M8 Communications; M9: Docking Functions

We present the implementation of one of the modules, M1, for the reader, the function F1 that is to realize M1. To define the thrust control there are a number of parameters that come to play that are hardware implied data or functionality related requirements. These are to be specified and then implemented by AI agents. Each function defined on the object corresponding to M1 is specified with a set of preconditions that imply coobjects to be defined for exception and recovery if the preconditions fail.

Object:= Thrust Control = <TCN, TCF> corresponding to normal and fault coobject

Ops:= Throttle Level Up (TLU) I Throttle Level Down (TLD)

Preconditions to be defined are on the following objects or parameter set (PS): {velocity, acceleration, attitude and tilt level, trajectory, control panel, obstacle encounters}, An operation TLU on the object M1, thrust control = <TCN,TCF>, is preconditioned by the above parameters that are implemented by functions that are defined on the coobject TCF to check for faults and exceptions. For example, when defining TLU the designer must keep in mind a velocity preconditions relative to the various parameters, that must be realized by a function defined on the coobject.

That function, let us refer to it by velocity check (vc) is a process that always checks the velocity to make sure it does not exceeds limits. Similar fault functions are defend for all the parameters in the set PS. Thus on the object TCN there is an operation TLU and on the coobject TCF there is an operation vc to implement velocity check preconditions for normal functioning of TLU. Each of the functions defined on the object or coobject are implemented by many agents. For example, when the velocity check function is invoked one or all of the following set of functions get activated.

An agent vca1 is invoked to signal M1 to activate some level of TLD, while checking with agents running off of panel and sensor readers. It also signals flap controllers if space craft is at stage where flaps are effective, to try to recover from hazardous conditions. If all fails it activates agents that are to implement automatic thrust control to bring velocity to acceptable levels. Similar set of agents are implementing for acceleration, attitude and tilt levels, trajectory violation, and obstacle encounters. In each case attempt is made by the coobject functions and their implementing agents to recover such that the precondition to an operation on the object is met. In the figure below objects are represented as <object,coobject>, where the coobject is a copy of the object on which faults and recovery functions are defined. The functions are implemented by agents ai, where ai are agents implementing velocity, acceleration, obstacle avoidance, or agents that check for limitations of such functions.

5. FTS SYSTEMS AND CONCURRENT FAULT CHECKS

The above multi-agent realization of the specifications implies design with a pair of concurrent systems. Each of the two systems is to be designed with a collection of kernels, such that there corresponds a kernel for each specification. A kernel consist of the minimal set of processes and objects that can be used as a basis for defining a computing activity. This term is analogous to the terminology familiar in operating systems concepts. The objects and the operations of one set of kernels once defined specifies the FTN, while those of the FTF are defined by the dual kernel. The set of kernels defining FTN and FTF are synchronized by cross operations and interact by some operations that are implemented by message communications between FTN and FTF. These operations are defined to either inform the various processes that are mutually dependent or to take the system from an active state in FTN to an active state in FTF. Note that when exceptional conditions occur the active state is FTF. However, both collection of kernels are considered concurrently "running."

FTF's major task is that of fault handling and recovery. If fault recovery takes place, in each kernel, the active kernel (a collection of agents) for a particular function, will be the FTN component, while the FTF component does concurrent checks for further exceptions should they be encountered. In each of the kernels there are objects, processes defining the operations, and objects to which there is a corresponding function in the other kernel. Thus FTN and FTF are a collection of objects and processes. FTN := <{O1,<p1,...,pn>},{O2,q1,q2,..},...{On,...},RNA>

RNA is the set of relations on each object and cross objects.

FTF := <{O1,<e1,...,en}>,{O2,<e11,e12,...,e1m>},...,{On,<...>},RF>

RFA is the set of relations on each objects and cross objects.

Each of the processes can have a corresponding agent in the dual family. The <FTN,FTF> pair in a computing system "run" as a concurrent family of processes. Various functions in FTN and FTF are represented by agents that are mutually informable across the <FTN,FTF> pair. The overall functionality of the system depends on the messages passed across from one agent to another. To each specification defined by FTKA there corresponds two kernels running concurrent. This is depicted by the following transition diagram.

6. MULTI AGENT DESIGN AND AGENT MORPHISM

In [12] we present new techniques for design by software agents and new concepts entitled Abstract Intelligent Implementation of AI systems (AII). The stages of conceptualization, design and implementation are defined by AI agents and Mediators[15]. Multiagent morphisms are proposed to facilitate software agent design. Objects, message passing actions, and implementing agents are defined by syntactic constructs, with agents appearing as functions. The proposed AII techniques provide a basis for an approach to automatic implementation. Interporatability is defined by mediators implemetening objects and agents. AII techniques have been applied to Heterogeneous KB[13] Design and implementation[14]. The application areas include support for highly responsive planning. Intelligent implementation of software, i.e., the design and implementation by AII techniques is due to be an area of crucial importance as the AI techniques are applied gradually to the real problems encountered in fields such as intelligent systems, aerospace, AI for robots, and various applications. AI systems might be defined by the stages of Conceptualization, Design, and Implementation. Flagrant Agent Computing by active agent learning and includes exception knowledge as an essential component an is treated in [14]. The techniques are defined for designing heterogeneous software. System implementation is by independent concurrent computing agents. AI and software systems are defined in the present paper by a pair of systems, each consisting of many computing agents. The initial phase of the design of the proposed AII techniques is to present the design with Mediator [14]. At the knowledge learning phase the expert is to state all exceptions to actions and what recovery and corrective actions are to be carried out. For each action on an object a dual action is to be supplied through FNK, such that a specifier can fully define the effect of the dual actions. As an illustration the following trivial example is presented.

```
Object:= Coffee_Constellation
OPS:= Serve_Coffee (Type,Table_no) | ......
Serve_Coffee (Spectacular_Brew,n) => Signal an available robot to fetch and serve
(Spectacular_Brew,table n)
Exp:= Serve_Coffee (Angelika,Table_no) |...
Serve_coffee(Angelika,Table_no) => if out_of_Angelika notify Table_no;
                                   offer cookie <and make use of
                                   intelligent decision procedures to
                                   offer alternatives>
```

APs := <A trivial example>, many robots appear at a critical entrance at once, necessitating FA activity. The above figure is an example mediator instantiation for the Stellar Robot Populated Coffeeshop. The present approach, once its theoretical basis is fully developed as defined in part by the present paper, consist of a pair of complex algebras, connected only by agent message passing. This leaves us an exciting prospect for a theoretical development of the <O,A,R> algebras and that of the A.I.I. theory. <O,A,R> is a pair of algebras, <Alg[A],Alg[F]>, connected by message passing and A.I.I. defines techniques for implementing such systems. New multiagent computing techniques were defined at our paper in [17].

7. CONCLUDING COMMENTS

The approach to multi agent software design presented here has actually been applied to challenging practical problems in system design by the present author. Illustrating examples are provided to clarify the design methods. The specific techniques have been a subject of research by the present author over the lst few years . The paper presented in [10,12] were the start in a series of concept papers crystallizing the field *Fault Tolerant Artificial Intelligence Systems*. Faults and exceptional behavior are an essential part of a system and must be thought of at conceptualization time. multi-agent implementations is the methodology to be applied to the design of the AI systems of the future. The abstract intelligent implementations techniques are further defined and developed by this author in [11,12]. Mutliagnet robot supervsion is yet another application[16].

REFERENCES

1. Nourani, C.F.,"Abstract Implementations and Their Correctness Proofs," JACM, April 1983.

2.Nourani,C.F.,"Planning and Plausible Reasoning in AI: Diagrams, Planning, and Reasoning," Proc. Scandinavian Conference on AI, May 1991, 150-157, Denmark, IOS Press.

3. Nourani,C.F.,"Multi Agent AI Techniques for Design and Implementation With Abstract Requirements, May 19, 1993, Process Coordination Wortkshop, August 1996, Kaiserslautern, Germany.

4.Nourani, C.F. and K.J. Lieberherr, "Data Types, Direct Implementations, and Knowledge Representation," Proc. 19th HICSS, Honolulu, Hawaii, January 1986, Vol II, pp. 233-243.

5. Siewiorek, D.P., "Fault Tolerance in Commercial Computers, Computer V23, no.7, July 1990.

6. Genesereth, M.R. and Nilsson, N.J., Logical Foundations of Artificial Intelligence," Morgan-Kaufmann, 1987.

7. Nourani, C.F.' "Parallel Module Specification'" November 1990, ACM SIGPLAN Notices, January 92.

8. Kelly, J.P., McVittie, T.J., Yamamoto, W.I., "Implementing Design Diversity To Achieve Fault Tolerance, IEEE Software, V8, no.4, July 1991.

9.Flavia, C. "Understanding Fault-tolerant Distributed Systems," CACM, V34, no.2, Feb. 1991.

10. Nourani, C.F. "A Multi Agent Approach To Fault Free and Fault Tolerant AI," Proc. FLAIRS-93, Florida AI Symposium, April 1993.

11.Nourani, C.F.,"Multi Agent A.I. Techniques For Validation and Verification," 1992, ACM SIGSOFT, February 1996.

12 Nourani,C.F."Abstract Implementation Techniques For AI Systems By Computing Agents: A Conceptual Overview," SERF-93, Orlando, FL., November 11, 1993.

13 Koopman, M.R.J., Spee J.W., Jonker W., Montero, L., O'Hara K., Mepham, M, Motta E., in't Veld L.J., VITAL Conceptual Modelling, VITAL Deliverable DD213, PTT Research, Groningen, December 1991.

14 Nourani, C.F.,"Abstract Intelligent Design Techniques for Heterogenous Systems," May1995, MAAMAW, April 1997, Ronneby, Sweden.

15 Gio Wiederhold: ``Interoperation, Mediation and Ontologies"; Proc.Int.Symp. on Fifth Generation Comp Systems, ICOT, Tokyo, Japan, Vol.W3, Dec.1994, pages 33-48.

16 Nourani,C.F., " Multi Agent Robot Supervision, January 10, 1995, Third European Workshop on Learning Robots, Heraklion, Crete, Greece, April 1995.

17. Nourani,C.F. "Slalom Tree Computing, A Computing Theory For Artificial Intelligence," AI Communcations, December 1996, IOS Press Amsterdam.

Information Systems Integration: Some Principles and Ideas

Nahla Haddar[1], Faïez Gargouri[1], Abdelmajid Ben Hamadou[1] and Charles François Ducateau[2]

[1] LARIS Laboratory, University of Sfax FSEG. B.P.1088, Sfax 3018.
[2] UIT de Paris, Laboratoire LIASI, 143 Avenue de Versailles 75016 Paris-France.
Abdelmajid.Benhamadou@fsegs.rnu.tn

Abstract. In this paper we present an approach to model a complex information system based on the integration of existing information systems. We emphasise on the comparison phase of the integration process in which similarities and conflicts between the initial systems must be detected. We show how object classes can be compared using behavioural aspects of objects and we discuss conflicts that can occur between object classes and we give rules to resolve them.

1 Introduction

The design of a complex information system (IS) is hard to be carried out by a single designer. Rather, it demands the participation of designer teams who may work separately and use possibly different design methods. In this way, the global IS becomes an assemblage of units (subsystems) endowed with various resources. But without coordination between these units, the global IS can not be used in practice.

Many approaches have been proposed to coordinate information systems work [2], [7]. For instance, the method proposed in [7] consists in constructing an agents system specialised in information sharing and coordination between systems. Although this approach is promising, it seems to us nevertheless too formal and requires the elaboration of a number of protocols between agents which increases exponentially with the number of agents.

Our proposition to coordinate the different parts of an IS is to integrate their conceptual representations. IS integration is a difficult task since the final IS has to satisfy the following properties of:

- Correctness: The final IS has to contain all concepts present in any information subsystem correctly.
- Minimality: If the same concept is represented in more then one component subsystem, it has to be represented only once in the final IS.
- Understandability: The integrated IS must be easy to understand for the user.

By taking these suggestions into account, we decompose the integration process into three steps: the initial IS sub-parts translation using a unification model, the comparison of the subsystems in order to identify the common elements between them, and finally the merging of subsystems into a global IS. In this paper, we study only the comparison step which is the most important step in the integration process.

The paper is structured as follows: section 2 is devoted to present the different steps of the integration process. In section 3, we describe the comparison step and give some formalisms to compare IS elements. Section 4 presents our conclusions and future works.

2 Information Systems Integration Process

The integration problem in the IS modelling and design field can be compared to schema integration in databases [3], [8], [9] on a higher abstraction level. Indeed, in the first case we start the integration from a set of conceptual representations, while in the second case the starting point is a set of existing databases.

Our integration process is carried out in three steps:

- Preintegration: When dealing with information subsystems modelled with different design methods, it is necessary to make their conceptual representations homogeneous to be able to compare and then to merge them. Thus, a framework is needed to unify the concepts used by the different analysis and design methods and to help the cooperation of heterogeneous modules and their reuse. The framework that we use is a object oriented generic model called MGCO2 [5], [6].
- Comparison: In order to obtain a minimal IS after integration, we have to identify the elements (attributes, classes, methods,...) which are common to the different subsystems. Therefore, we need to determine the correspondences between these elements. This task is far from being trivial especially when the number of the involved subsystems is significant.
- Conflict resolution and merging: In this step, the conflicts detected in the previous step must be resolved. Then, the subsystems merging can begin. This step gives birth to the final IS conceptual representation. It is this representation which will be implemented.

Figure 1 shows the different steps of the IS integration process [6]. In this figure, different sub-systems IS1, IS2, ..., ISn are modelled using analysis and design methods M1, M2, ..., Mn. After the preintegration step, each sub-system is modelled in the Unification Model (UM).

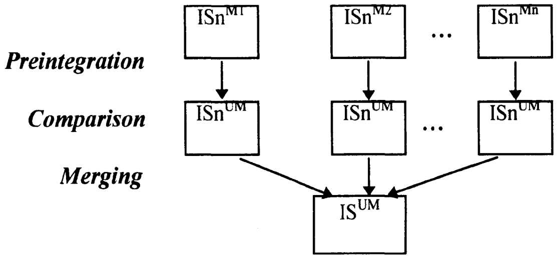


Fig. 1. The three integration steps

3 Comparison Step

Once the conceptual representations translation is done, the next step in the integration process is to find common elements between the original subsystems. The real word semantic of a class is given by the attributes, the methods and the state graphs describing the class objects and their behaviour. It follows that we can compare classes by comparing attributes, then methods and finally state graphs.

3 . 1 Example

Let's take the example of a complex application 'Clinic' concerning the data management of a clinic. It is decomposed into many sub-applications from which we distinguish the two sub-applications 'Administration' and 'Surgery' concerning respectively the management of information of the administration and those of the general surgery department.

```
Class Person: Abstract
Properties:
    number: INTEGER
    f_name: STRING
    name: STRING
    age: INTEGER
    address: Address
 Static constraints:
    Uniqueness: number
 Methods:
    create()
    destroy( )
 ENDCLASS Person
Class Address: Abstract
 Properties:
    street: STRING
    nr: INTEGER
    city: STRING
Methods:
    change_adr()
ENDCLASS Address
Class Patient: Concrete
Inherits Person
 Properties:
    dateadm: DATE
    service: STRING
 Methods:
    change_ser()
ENDCLASS Patient
Class Employee: Concrete
Inherits Person
Properties:
    entrance_date: DATE
    salary: INTEGER
 Methods:
    change_sal()
 ENDCLASS Employee
 Class Physician: Concrete
 Inherits Employee
 Properties:

        specialty:
STRING
ENDCLASSPhysician
```

Fig. 2. Textual representation of S1

```
Class Operation: Concrete
Properties:
    nr: INTEGER
    op_date: DATE
    responsible: Physician
    patient: Patient
Methods:
    create()
    destroy()
    change()
ENDCLASS Operation
Class Person: Abstract
 Properties:
    number: INTEGER
    f_name: STRING
    name: STRING
    age: INTEGER
    address: aggregation of
          {street: STRING,
          nr: INTEGER,
          city: STRING}
Static constraints:
    Uniqueness: number
Methods:
    create()
    destroy( )
    change_adr()
ENDCLASS Person
Class Patient: Concrete
Inherits Person
Properties:
    adm_date: DATE
Methods:
    change_state()
ENDCLASS Patient
Class Physician: Concrete
Inherits Person
ENDCLASS Physician
```

Fig.3. Textual representation of S2

The class S1.Physician represents all the clinic physicians while that of S2 represents only surgeons. Besides, the class S1.Patient represents all in the clinic admitted patients and S2.Patient represents only patients who undergo an operation.

3 . 2 Real World Semantic

Although an IS represents objects of the real world, with their properties and their behaviour, however the integration process exceeds representations to consider first what is represented (the semantic aspects) rather than how it is represented (the static aspects) [9]. For example, we want to know if the class S1.Person is the same as the class S2.Person even if the properties of the two classes are not identical. Thus, we say that two information systems have some common things, if objects of the real world that they represent have common elements. The determination of correspondences between elements of information systems is therefore based on the real world objects semantics. We define the real world semantics (RWS) of an object class as being the set of all real world objects which properties and behaviour are those represented by this class. Our definition of the RWS is different from that proposed by [3], [8], or [9] concerning database schema integration in the fact that the latter is based on the class extension occurrences.

3 . 3 Attribute Comparison

The attribute comparison is based on the comparison of class structures i.e. the class attribute names and their types.

Let S be an IS and C a class of S. Each attribute a of C is defined by its name N(a) and its type t(a). So Att(C) is the set of couples defining the C attributes.

$$Att(C)=\{(N(a), t(a)) \,/\, a \text{ is a C attribute}\}.$$

A set of class attributes is formed by the attributes inherited from the superclasses and the specific class attributes. Before comparing two classes, it is therefore necessary to determine all their attributes. In order to formalise this, we formalise first the 'is-a' relation between classes. The formalism which we use is similar to [11].

Definition 1. The subclass relation on S, denoted by sub(S), is defined as the reflexive and transitive closure of :
$isa(S) = \{(C1, C2)/ C1, C2 \in S \wedge C2 \text{ is a superclass of } C1\}$.
Now we can define the set of attributes of an S class C .

Definition 2. Let A be the set of the C specific attributes. The set Att(C) of all attributes of C is defined as $Att(C) = A \cup \{ (N(a), t(a)) \mid \exists C' \in S: (C, C') \in isa(S) \wedge (N(a), t(a)) \in Att(C') \wedge \forall (N(a'), t(a')) \in A: a \neq a' \}$.
Every class in an IS has a type which describes the its structure. Informally, the structure of a class is an aggregation of its attributes.

Example.
In our example we have: Att(S1.Person)= {(number, INTEGER), (name, STRING), (f_name, STRING), (age, INTEGER), (address, Address)} and Att(S2. patient)= {(number, INTEGER), (name, STRING), (f_name, STRING), (age, INTEGER), (address, aggregation_of{street: STRING, nr: INTEGER, city: STRING.)}

The structure of a class can be represented by an infinite labelled tree whose nodes are types and whose leaves are predefined types. The edges labels are attribute names. The relationship between two class types is then expressed in terms of homomorphism between trees representing their structures. From this mapping, the common attributes between classes are deduced. A detailed study of this formalism can be found in [11].

Example. The Structures of S1.Person and S2.Person will be represented by the two trees given by figure 4. Note that the trees are not infinite because we have no recursive types.

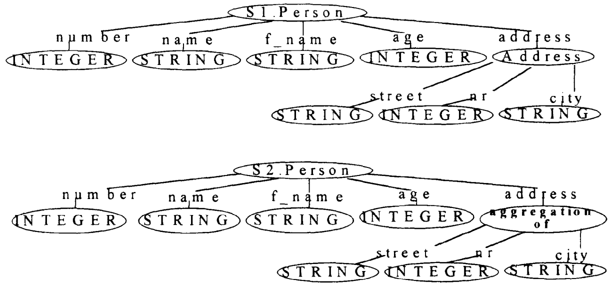


Fig. 4. Structure of S1.Person and S2.Person

The two trees are isomorphic since there is a bijective map between their nodes and edges. Consequently type(S1.Person) and type(S2.Person) are equivalent and the two classes have the same static structure and consequently they share the same attributes. □

The attributes of a class represent its static aspect. Class methods are part of its dynamic aspect. If two classes share a significant number of attributes, they will probably share some methods which use some of the common attributes. The method comparison is presented in the next sub-section.

3 . 4 Methods Comparison

Each method m of an object class has a body b(m), a type r(m) (result provided by the method) and a signature sig(m) giving the name and the set of attributes. Meth(C) is then the set of triplets defining the methods of a class C.

Meth(C)={(sig(m), b(m), r(m)); m is a C method.}

The methods set of a class is formed by the methods inherited from the superclasses and the specific class methods.

Methods of a class can be modelled using so called class-methods whose attributes represent the arguments, the body and the type of a method. (The name of the method is given by the name of the class). This concept is used also in Shood [4].

Links between the class-methods are semantic links between the different methods. A definition of the different types of links is given in [10]. Especially, one distinguishes the inheritance link which models an overload. Indeed, an inheritance link between a super-class-method X and a sub-class-method indicates that the method Y overloads the method X.

So, similarly to class structures defined in the previous sub-section, we can define structure of class-methods. Thus, comparison of methods of two classes of different information systems leads to attribute comparison of class-methods. The difficulty here resides in the comparison of method bodies. From the theoretical point of view, this comparison is possible. By choosing an appropriate formalism to specify method bodies this comparison can also be carried out automatically [11].

A class state graph describes the behaviour of its objects. This behaviour is expressed by the set of methods changing an object from a state to another. Thus, if two classes have some common methods, it is probable that they share also some parts of the state graphs of each other. So, the study of the relation ship between state graphs is important.

3 . 5 State Graph Comparison

A class state graph describes the behaviour of its objects in response to an event occurrence. This behaviour is the same for all the class objects. Basing on this property, we try to translate semantic relationships between objects of two classes belonging to different information systems. Each link will be expressed in terms of relationship between the state graphs of the considered classes. This helps, given two state graphs, to determine the relationship existing between the corresponding classes. This relationship can be either an inclusion, a strict intersection, or a disjunction.

Definition 1. Let C be a class of an IS S, and O an object of this class. Let st(C) be the set of states that an object O can take during its life cycle LiC(O). st(C) contains a particular state state0 that corresponds to the state taken by the object O before its creation and after its destruction. The C state graph, G(C), can be defined as follows:

$$G(C)= (Evt(C), Const(C), Meth(C), st(C), state0,\delta, \lambda)$$

where the functions δ and λ are defined by:

δ:st(C)×Evt(C)×Const(C) → st(C)

δ(q, e, c) = the state taken by an object of C being in the state q in response to an event e when the condition c is satisfied.

λ:st(C)×Evt(C)×Const(C)→ Meth(C)

λ (q, e, c) = the action activated on an object of C being in the state q in response to an event e when the condition c is satisfied. □

Example. The state graphs of S1.Patient and S2.Patient are represented in figure 5a and figure 5b respectively.

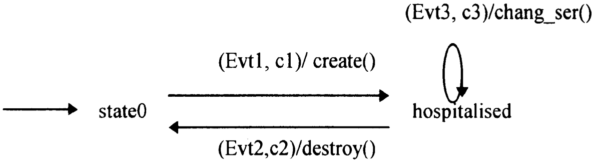


Fig. 5a. State graphs of S1.Patient

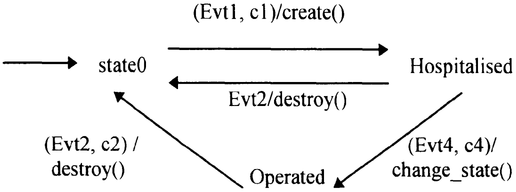


Fig. 5b. State graphs of S2.Patient

where Evt1: appoint patient to a service;
Evt2: patient leaves the clinic;
Evt3: patient changes services;
Evt4: operate patient.
c1, c2, c3, c4 are constraints.

Definition 2. A path p(state0, qn) in G(C) is a transition sequence (state0, (e0 , c0)/ m0, q1)(q1, (e1 , c1)/ m1, q2) ... (qn, (en , cn)/ mn, q{n+1}), where n>0 and for all i, $0<i<n$ we have: δ(qi, ei, ci)= q{i+1}.
P(C, qn) is the set of paths p(state0, qn) in G(C).
The length len(p(state0, qn)) of a path is the number of transitions that compose it.
The event sequence relative to p(state0, qn) is e0 e1... en.
The action sequence generated by p(state0, qn) is m0 m1... mn.□

Remark that the event sequence relative to a path p(state0, qn) in G(C) represents a scenario of the life cycle of an object of the class C.

In the following, we study the relationship that can exist between two state graphs.

Definition 3. Let S1 and S2 be two distinct information systems, C1 and C2 be two classes belonging respectively to S1 and S2.
$G(C1) \subseteq G(C2) :\Leftrightarrow Evt(C1) \subseteq Evt(C2) \wedge st(C1) \subseteq st(C2) \wedge Const(C1) \subseteq Const(C2) \wedge Meth(C1) \subseteq Meth(C2) \wedge$ $\delta 1$ and $\lambda 1$ are the restrictions of $\delta 2$ and of $\lambda 2$ to the set $st(C1)\times Evt(C1)\times Const(C1)$ respectively. □

If the objects behaviour of a class is the same as a part of the objects behaviour of another class, then the former contains the first. This fact can be demonstrated by the following lemma.

Lemma 1. $G(C\ 1) \subseteq G(C\ 2) \Rightarrow C1 \subseteq C2$. □

Proof. Suppose that $G(C1) \subseteq G(C\ 2)$. Then $P(C1, state0) \subseteq P(C2, state0)$. If $C1 \not\subset C2$ then one could find an object O such that $O \in C1$ and $O \notin C2$. Since $O \in C1$ means that there exists a path p(state0, state0) in P(C1, state0) whose sequence of events is a scenario of LiC(O) and $O \notin C2$ means that no path in P(C2, state0) corresponds to a scenario of LiC(O), we obtain a contradiction with our hypothesis because p(state0, state0) belongs to P(C1, state0). □

Definition 4. $G(C1) \cap G(C2) \neq \phi$: $\Leftrightarrow$ $Evt(C1) \cap Evt(C2) \neq \phi \wedge st(C1) \cap st(C2) \neq \phi \wedge Const(C1) \cap Const(C2) \neq \phi \wedge Meth(C1) \cap Meth(C2) \neq \phi \wedge \forall$ p(state0, state0) $\in$P(C1, state0), len(p(state0, state0))>1: p(state0, state0)$\in$P(C2, state0). □

If the behaviour of a subset of class objects is the same as a subset of objects of another class, then the two classes have a non empty intersection. This fact be demonstrated by the following lemma. □

Lemma 2. $G(C1) \cap G(C2) \neq \phi \Leftrightarrow C1 \cap C2 \neq \phi$. □

Proof. $G(C1) \cap G(C2) \neq \phi \Leftrightarrow Evt(C1) \cap Evt(C2) \neq \phi \wedge st(C1) \cap st(C2) \neq \phi \wedge Const(C1) \cap Const(C2) \neq \phi \wedge Meth(C1) \cap Meth(C2) \neq \phi \wedge \forall$ p(state0, state0) $\in$P(C1, state0), len(p(state0, state0))>1: p(state0, state0)$\in$P(C2, state0). $\Leftrightarrow \exists O$: the event sequence relative to p(state0, state0) is a scenario of LiC(O) $\wedge$ $O \in C1 \wedge O \in C2$ $\Leftrightarrow$ $C1 \cap C2 \neq \phi$. □

Thus, in order to compare two classes from different information systems, we try first to compare their attributes and methods. Once some similarities are detected, we compare the state graphs.

3 . 6 Conflicts Taxonomy

When a correspondence describes some elements as being identical (i.e., they have the same representation and the same semantics) their integration is then obvious: the integrated element (which will be present in the final IS) will be identical to the input elements. But, in most cases, the corresponding elements present some differences in their representations or in their semantics. This case leads to a conflict situation. We give hereafter a taxonomy and some examples of conflicts occurring when comparing two subsystems.

- *Classification conflicts*: A classification conflict occurs when the corresponding classes describe object sets which are different but semantically linked. In our example Patient in S1 describes patients that have been admitted in the clinic while Patient in S2 describes only patients who undergo operations.
- *Structural conflicts*: A structural conflict occurs when the corresponding elements are described using different concepts belonging to different abstraction levels, for example a class and an attribute. In our example, there is a structural conflict between the S1 class Address and the S2 attribute Person.address.

Other types of conflicts can be met, namely, descriptive conflicts and dynamic aspects conflicts.

3 . 7 Conflict Resolution

Once the correspondences between systems are established, the integration can begin. Every correspondence is analysed in order to determine which integration rule will be applied to obtain the corresponding final IS elements. The major difficulty of this step to resolve is the emphasis on the different conflicts and the semantic problems detected in the comparison step and their resolution. Some merging rules must be therefore established .

As a solution for the classification conflicts we propose to include in the integrated IS an appropriate generalisation-specialisation hierarchy as detailed by the following rules. The resolution of the other conflicts requires a deep study of the abstraction level of the concerned elements.

Rule1: $G(C1) \subseteq G(C2) \Rightarrow \text{Generalise}(C2) \wedge \text{Mask}(\text{Meth}(C2) \setminus \text{Meth}(C1)) \wedge \text{Mask}(\text{Att}(C2) \setminus \text{Att}(C1)) \wedge \text{Specialise}(C1, C2)$

Rule2: $G(C1) \cap G(C2) \neq \phi \Rightarrow \text{Generalise}(C1 \cap C2) \wedge \text{Specialise}(C1 \setminus C2, C1 \cap C2) \wedge \text{Specialise}(C2 \setminus C1, C1 \cap C2)$

where Generalise(C) is a method which creates a generalisation class C; Specialise(C, C') is a method which makes from C a specialisation of class C'; and Mask is a function defined on the set of attributes and methods of a class and it allows the masking of some properties of classes [1]. By comparing both systems of our example we obtain following similarities and conflicts:

- the classes S1.Person and S2.Person are equivalent because they have the same attributes, the same methods and the same state graphs;
- the classes S1.Patient and S2.Patient have a not empty intersection;
- the class S2.Physician is included in S1.Physician.

By applying the above defined rules on our example we obtain the following integrated IS conceptual representation:

```
Class Person: Abstract
Properties:
number: INTEGER
f_name: STRING
name: STRING
age: INTEGER
address: Address
Static~constraints:
  Uniqueness: number
Methods:
 create()
 destroy( )
ENDCLASS Person

 Class Address: Abstract
Properties:
 street: STRING
 nr: INTEGER
 city: STRING
Methods:
 changeadr()
ENDCLASS Address

Class Patient: Abstract
  Inherits Person
Properties:
dateadm: DATE
ENDCLASS Patient

Class Patient1: Concrete
Inherits Patient
Properties:
 service: STRING
Methods:
 changeser()
ENDCLASS Patient1

Class Employee: Concrete
Inherits Person
Properties:
 entrancedate: DATE
 salary: INTEGER
Methods:
 changesal()
ENDCLASS Employee

Class Physician1: Concrete
  Inherits Employee
Properties:
 specialty: STRING
ENDCLASS Physician1

Class Operation: Conc.
Properties:
 nr: INTEGER
 opdate: DATE
 responsible: Physician2
 patient: Patient
Methods:
 create()
 destroy()
 change()
ENDCLASS Operation

Class Patient2: Concrete
 Inherits Patient
Methods:
 change_state()
ENDCLASS Patient2

Class Physician2: Concrete
  Inherits Physician1
ENDCLASS
Physician2
```

Fig. 6. Textual representation of the integrated IS S

4 Conclusion

In this paper we have given a cooperative information systems modelling method based on the integration of the systems conceptual representations. We have described only the main step of the integration process, namely the comparison step. The novelty of our approach is the use of state graph relationships to determine correspondences between classes. The theopretical approaches used to compare classes are invariant w.r.t. renaming of attributes or methods.

Further research includes improvement of all the discussed theoretical aspects used to compare information systems elements. We are also studying the different possible conflicts between information systems elements and trying to extend the set of merging rules.

References

1. Castellani, X.: MCO: Méthodologie générale d'analyse et de conception des systèmes d'objets Tome1: Ingénierie des besoins. Edition Masson (1993).
2. Ducateau, C. F., and Picavet, M.: Progressive adjusting Process for Data Modeling. In Proceedings of the International Conference on Industrial Engineering and production Management. Maroco (1995) 235-244.
3. Dupont, Y.: Problématique de résolution contextuelle des conflits de fragmentation dans l'intégration de schémas. Ingénierie des Systèmes d'Information Vol. 3-N°1 (1995) 29-58.
4. Escamello De Los Santos, J. G.: Shood, un modèle méta-circulaire de représentation des connaissances. Ph.D. diss., Institut National Polytechnique de Grenoble, France (1993).
5. Gargouri, F.: Proposition d'un modèle générique pour la conception orientée objet des systèmes d'information et étude de son implantation. Ph.D Paris 5 University (1995).
6. Gargouri, F., Ducateau, C. F., Gargouri, W. , Haddar, N.: About cooperation for complex applications and information systems modeling. In Proceedings of the International Conference on Industrial Engineering and Production Management. Lyon, France (1997) 258-267.
7. Koriche, F.: Une méthode de modélisation des systèmes d'information coopératifs. Ingénierie des Systèmes d'Information. Vol.4 -N°2 (1996) 195-218.
8. Larson, J. A., Navathe, S. H., Elmasri, R.: A theory of attribut equivalence in databases with application to schema integration, IEEE Transactions on Software Engineering. Vol. 15-N°4 (1989) 449-463.
9. Parent, C., Spaccapietra, S.: Intégration de bases de données: panorama des problèmes et approches, Ingénierie des Systèmes d'Information. Vol. 4-N°3 (1996) 333-358.
10. Ravat, F.: La fragmentation d'un schéma conceptuel orienté objet, Ingénierie des Systèmes d'Information. Vol. 4 -N°2 (1996) 161-193.
11. Thieme, C., Siebes, A.: Guiding schema integration by behavioral information, Information Sytems. Vol. 20-N°4 (1995) 305-316.

An Emergent Paradigm for Expert Resource Management Systems

Swamy Kutti
nsk@deakin.edu.au
School of Computing & Mathematics
Deakin University, Victoria
Australia – 3217

Brian Garner
brian@deakin.edu.au
School of Computing & Mathematics
Deakin University, Victoria
Australia - 3217

KEYWORDS

Expert system, resource management, semantic map, paradigm, heuristic map, meta-knowledge.

Abstract Data structuring mechanisms suited to complex problem solving environments have, in general, only been satisfactory for static processes. The challenge of complexity, for example, of process dynamics, in which human managers interact continuously with control systems and environmental changes, has focussed our research program on new paradigms in knowledge engineering and their associated requirements for novel data structures to support knowledge modeling.

This paper reports our conclusions regarding an emergent paradigm for expert resource management systems and identifies the requisite data structure, which is based on a semantic network abstraction "system map". Other examples of emergent paradigms in knowledge engineering will be found in [Garner, 96]

1 Introduction

In general, the more common decision making process published in the literature [1] are based on the heuristics of people controlling such processes. Such processes have provided suitable problem solving environments for the most frequently cited expert systems (e.g., medical diagnosis, fault diagnosis in machines, etc). The heuristic in such first generation systems were computationally tractable due to their implementation in program form using simple notations for rules and attributes. Basically, the programs compare instances of data abstractions and map to possible solutions using hierarchically classified heuristics. However, while rule-based data abstractions solve simple problems of a linear nature, there are some serious disadvantages in extending the use of this approach; namely:

- To add any new heuristic to the existing classification is a somewhat tedious (manual) process; sometimes the whole rule pattern may have to be reorganized!;

- Analysis of the abstracted data is purely sequential, which degrades the overall speed of solution discovery. For a growing expert system, this factor will be a major bottleneck;

- Data abstraction is normally dependent upon some specific language rather than a conceptual (generic) form. This feature limits heuristic processing, due to the functional characteristics of the selected language;

- Problem decomposition is difficult for dynamic environments, and hence, expert systems implementation for non-trivial decision making processes (specifically, real-time management problems) has been found to be difficult in practice. Refer to [10].

Work at Deakin on both a *Knowledge-based kernel for distributed operating systems* [4], *Information Systems Models for creative processes* [12] and *Heuristic Map* [3] has identified sources of new heuristics, resulting in the development of a semantic model for extensible heuristics, based on an object-oriented data abstraction. In modelling expert systems for resource management, a semantic network is expected to describe the structure of management control and associated attributes very precisely, supported by the concept-type hierarchy [8]. The concept-type hierarchy helps to establish and maintain a management regime with instances of managerial concepts and resources, and their inherited properties, acting as necessary parameters for scheduling operations. In other words, a complete network of concept instances is required to emulate a suitable data structure for resource management. Since the number of concept types and their instances are finite for a particular domain, a resource management semantic network is achievable. More precisely, this new data model has an *object-oriented semantic network* as its underlying data structure, emulating a hierarchical management model. The high level functions are configured in the form of "manager objects", represented by the object-oriented nodes of this network. Each "manager object" has the ability to gather and maintain sufficient peripheral knowledge to carry out its own resource management functions. This new model is called SYSTEM MAP as it represents the total system capabilities. SYSTEM MAP is itself an object that takes responsibility for the overall system performance.

In all combinatorial problems (i.e. in most resource management) the convergence of an expected solution can be realized when heuristics are applied from higher level objects to lower level objects. When a lower level object cannot find a solution, it can interact with its immediate higher level object to resolve conflict. This paper explains how this abstract data structure (i.e. SYSTEM MAP) may be used to derive a generic model called HEURISTIC MAP, in which meta-level heuristics, with cooperative feedback between levels can be applied to resolve combinatorial scheduling conflicts.

Further investigation of this heuristic abstraction has resulted in the development of a novel data structure that accommodates both hierarchical and peer-coupled heuristic classification. Basically, this paradigm accommodates both data abstraction and heuristic processing. It also implicitly represents conceptual graph structures [9]. The meaning of management control and types of control architectures are now described.

2. Management Models

2.1 Meaning of Management Control

A vital function of any management system is to maintain optimum control over the underlying resources in order to achieve a set of goals. In general, such resource management kernels are complicated and require a set of knowledge-based elements such as planning, scheduling and monitoring which are understood as essential features of any decision making process. These knowledge elements are basically of either an empirical or heuristic nature, thereby supporting traditional human experience in making decisions, in refining and storing knowledge, and in exercising their knowledge modeling skills.

Complexity of a decision making process depends upon the goals and the diversity of resource types involved in achieving these goals. Limited resources usually relate to simple goals and a single manager. Single (human) managers can handle such resource management with the required efficiency. On the other hand, corporate management within industries, government organizations, defence establishments, etc, usually have multiple goals, and hence have to manage vast resources of various types. Other related issues in complex resource management include:

- decision making in real-time,
- reliability of the desired control function,
- dynamic growth (extensibility)

Real-time decision making processes in particular, require not only the accumulation of sufficient knowledge relevant to a goal, but also a knowledge modeling process that allows the application of such knowledge in real-time. Secondly, the reliability of control functions depends upon proven methods in the application of knowledge. Meta knowledge, thus, becomes an essential component of the whole knowledge base. Thirdly, fault tolerance needs additonal heuristics to bring the system back to a normal state. Finally, dynamic growth depends upon a management control structure. It is essential to select a suitable control structure that allows the management process to expand dynamically without limitation.

2.2 Types of Control Architectures

There are several factors involved with the resource management process in adapting a suitable management structure. Such factors include the amount of resources, diversity of resources, diversity of goals, cost effectiveness, etc. In general, control in resource management systems requires one of the following strategies:

- Autocratic control
- Sequential control
- Partitioned control
- Co operative autonomous control
- Hierarchical control

A discourse on such control strategies is provided elsewhere [4].

2.3 Knowledge Distribution in Management Models

The centralized characteristic of decision making processes is exemplified by such strategies as *'Autocratic'*, and *'Autonomous but Sequential'* . Unless meta-knowledge is used to manage the knowledge base, a possibility of computational explosion exists. Reasoning in knowledge-based systems (including human managers) is defined as a process of accumulating information by inference until the solution to a given problem is identified. During reasoning, knowledge is usually processed in sequential mode, and exploring a large knowledge base for decision making will typically slow the overall performance. Moreover the centralized nature of most knowledge bases leaves little scope for interaction, and hence there is no way to justify the correctness of decisions made by computer. Finally, the knowledge engineering process is, overall, very cumbersome, due to the maintenance of a large centralized database.

In the case of partitioned control, domain knowledge exists at one level, but is partitioned according to specialization. The management is performed in a typical multitasking manner. That is, knowledge processing may be classified depending upon either resource types or functions and can be performed concurrently. This effectively improves the speed of the overall management process. This scheme is much faster than the centralized case. However, the reliability of the knowledge source is still poor. The reliability can, however, be improved by distributing knowledge sources to each partition. But there is a major disadvantage to this scheme: that is, the knowledge interaction is minimal due to the high degree of specialization of the partitioned knowledge source. Feedback is, thereby, constrained, resulting in less opportunity for management improvement of performance. This scheme basically uses the combinatorial nature of management analysis in achieving the solution. A major problem in this method is to synchronize the actions of the various knowledge sources in completing each management task required by the process. This scheme may be more suitable for production-oriented resource management, which usually requires less knowledge, as well minimal knowledge interaction using pipeline style processing.

In the case of the *hierarchical model*, the knowledge is abstracted vertically and distributed horizontally. The rationale behind the partitioning of the knowledge is to provide meta-knowledge with more heuristics at higher levels for resolving conflicts, and in turn, achieving goals such as performance tuning, load balancing, fault tolerance, etc, thereby leaving the highly specialized knowledge at the lower levels for production maintenance. Basically a heuristic can be used to build an inference engine at one level to control the actions of its lower level. Although heuristic knowledge is essential at every level, its distribution varies throughout the levels. That is, fewer heuristics at the lower level and most heuristics at the highest level. The variation of heuristics through out the levels should be linear for achieving well-tuned hierarchical management.

This strategy also embraces the requirement for distribution of heuristics within the partitioning of each vertical level, thereby improving the reliability and availability of the knowledge sources. Knowledge interaction exists both at peer level and at different levels. Vertical abstraction provides a basis to partition the knowledge at decision making level or provides a mechanism called meta-level reasoning, by which decision making can be hierarchically planned to solve even complex problems.

3 DYNAMIC DATA MODEL

3.1 Scope of Semantic Network in Resource Management

Semantic networks can be used to represent declarative knowledge expressed in graphical form, and they enable us to record information about observed real world objects. This concept was first introduced in Quillian's work [7] on natural language understanding. The basic idea is that the message content can be considered as autonomous at the conceptual level. Conceptual graphs derived from a discourse permit subsequent analysis of future dialog on related concepts. Though the initial work focused on developing a semantic network for machine understanding of natural language, many other expert systems adopted this semantic net mechanism as their domain knowledge representation scheme [6], [2]. The functional requirements of resource management systems represent a typical case where an object-oriented abstraction would considerably simplify the task of designing and maintaining these systems. The resource management in a distributed environment requires careful resource allocation for tuning the systems performance. Moreover, as growth occurs distributed resource management has to accommodate new resources dynamically. The resource management load tends to increase. This implies that the supervisory role in such systems becomes a major concern, and hence dynamic control management is necessary in addition to the lower level functions, which are more akin to process control. The complexity results from the various data structures necessary for the supervisory and resource management functions that are involved in supporting user processes and their needs. While many languages provide static data structures suitable for a uniprocessor operating system, it is very difficult to realize the diversified, functional environment of a distributed resource management system in practice.

Our new data model has an object-oriented semantic network as its underlying data structure emulating an hierarchical management model, and is termed a "SYSTEM MAP" (refer to **Figure 1**).

3.2 SYSTEM MAP STRUCTURE

In application domains such as resource management, the conceptual elements of the network are expected to emulate real-world entitles. The dynamic nature requires each concept 's instance to be associated with the necessary operational procedures to act on the surrounding structural and factual information. Each element of the net behaves like an *ACTOR*, or more generally, as an element of the abstract data structure, due to the distribution of knowledge. Hence, a semantic network of this type is a collection of these objects (or elemental abstract data structures) that are interconnected structurally and work in a coherent manner to maintain and exercise domain knowledge. The organization of the knowledge-base presents a view of an abstract mechanism that hides the information within the nodes or objects. Moreover, the hierarchical nature of the semantic network structure is ideally suited to the emulation of an application domain of the hierarchical type.

A domain graph is supported by two types of dictionaries, namely the object dictionary and the relation dictionary. The object dictionary is used to keep track of objects in the graph, to identify their types, to locate the object templates and to store the objects access modes. The global relation dictionary keeps the location of the relations in the whole domain graph. Each subdomain is accompanied by a domain dictionary for its objects (or concepts). The root object of each subdomain represents the domain object itself. Type hierarchy information is part of the data structure. The role hierarchy can maintain the manager hierarchy for resource management. The concept type hierarchy and accompanying templates may be used to construct the semantic network data structure.

Since a modular approach improves both design and processing efficiency, our domain graph has been structured into subdomains. Each subdomain has its own object dictionary. The graph maintenance functions may be incorporated in the root domain object of the network called *SYSTEM* to provide a user interface. Operations for a domain-based network include:

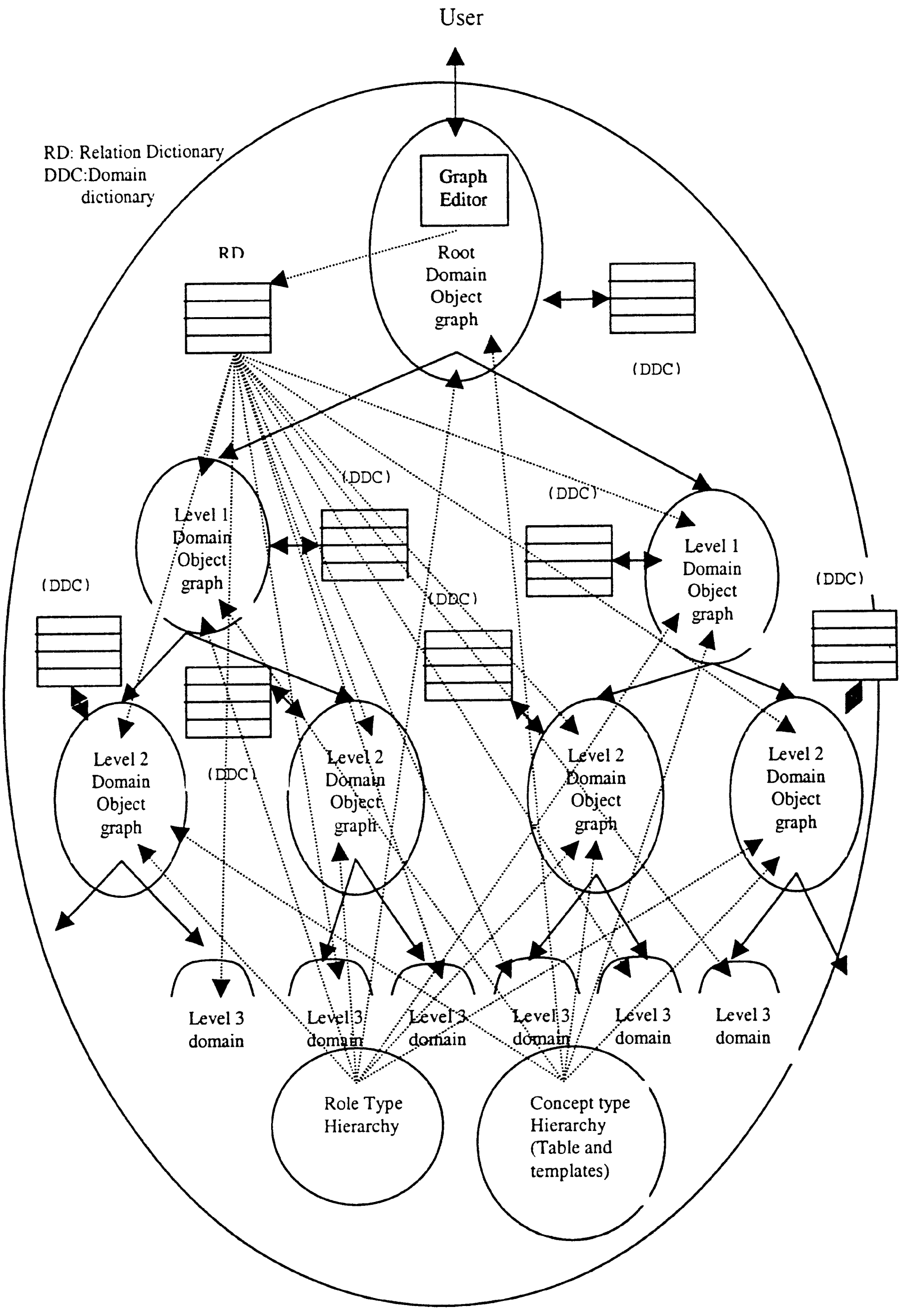


Figure 1: A dynamic data Structure Model based on Semantic Network
(SYSTEM MAP)

1. make template (or make concept type)
2. make domain concept.
3. change domain.
4. make assertion.
5. copy assertion.
6. restrict assertion.
7. join assertion.
8. simplify graph.
9. check assertion.
10 delete concept (or assertion).
11. show concept.
12. Activate and deactivate concepts
13.move concept.
14. locate domain.
15 list domain dictionary.
16. list relation dictionary.
17. check access rights of concept

4 EXPERT SYSTEMS MODEL FOR DISTRIBUTED MANAGEMENT SYSTEMS

4.1 Representation of Criteria for Decision Making

Examples of the domain specific concepts of a management system are manager entities themselves, their characteristic, their relationship with other manager entities, tasks associated with each manager entity, etc.

The systems should also be capable of representing the criteria for making decisions based on the domain knowledge. To carry out the actual process of intelligent managerial support, meta-level knowledge about the execution-mechanism needs to be represented in the system. The knowledge processing capability is supported in each abstract object by an additional data structure called the *skill-base environment.* This data structure provides a mechanism to support the goal specification, the procedures for reasoning and inferring, and an event mechanism to activate the object. SYSTEM MAP may be visualized as an object-oriented semantic network that maintains the distributed control structure to support the domain knowledge of resource management system. The system derives its intelligence and effectiveness from this knowledge, which in turn must be maintainable with ease.

The skill base component of the knowledge base is fully distributed among the manager objects and consists of a rule file and an associated inference engine. The rule files consist of rules in declarative form, and represents the heuristic knowledge of an object. The knowledge-base editor component helps a knowledge engineer to add/modify the rules or to add/remove manager objects, etc. **Figure 2** shows a control model of our expert system used in hierarchical management systems.

4.2 Heuristic Map

The systems knowledge base has three distinct components:

(i) knowledge dictionary/database
(ii) skill (actors) of the knowledge-base; and
(iii) knowledge-base editor/interpreter.

The database component of the knowledge-base maintains, basically, two types of information as follows:

(a) structural details of the system objects; and
(b) factual data showing the status of the objects environment.

The skill-base for resource management uses heuristics at every level to support inference. As an instance, an abstract manager object could accumulate new knowledge through *inference*, and then apply that new knowledge to adjust the system performance. Each higher level object applies its own heuristics to the partial solution provided by the heuristics of lower level objects. The invocation of heuristics can occur both in a bottom up flow or top down, depending on the level of control to be exercised, as shown in the heuristic map (**Figure 3**). At the top level a manager object maintains its heuristic knowledge stored as rules. Similarly, the objects at lower levels maintain their own heuristics in their respective rule files, which are then processed by inference mechanisms provided within the objects.

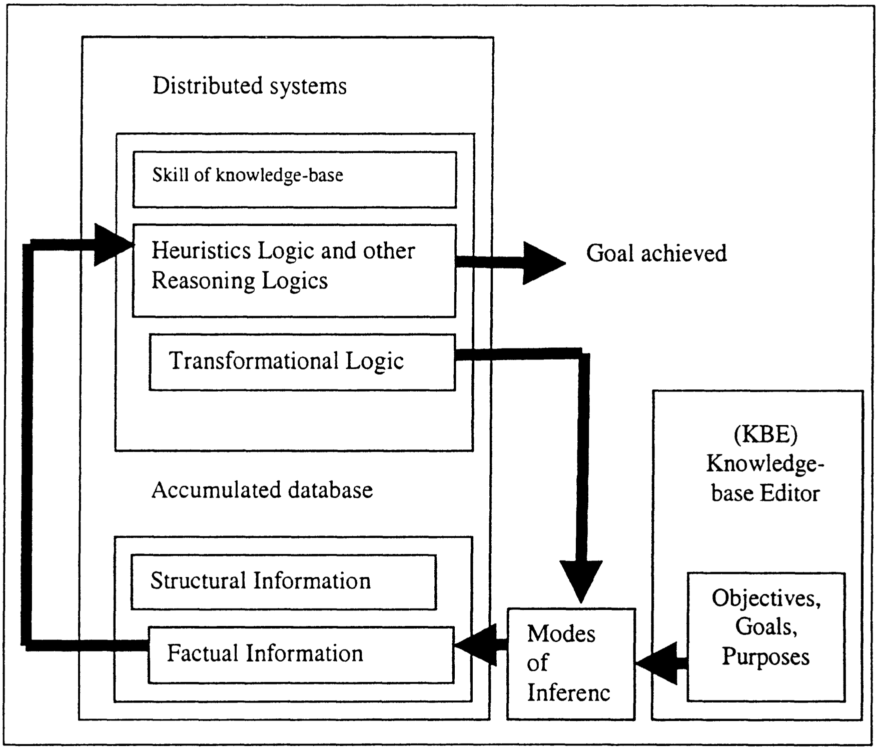


Figure 2: Expert System Model for Distributed Management Systems

In all combinatorial problems (i.e. most resource management problems) the convergence of an expected solution can be realized when heuristics are applied from higher level objects to lower level objects. When a lower level object cannot find a solution, it can interact with its immediate higher level object to resolve conflicts. This method of heuristic classification has already been demonstrated using an object-oriented semantic network in modeling a kernel of a distributed system [4].

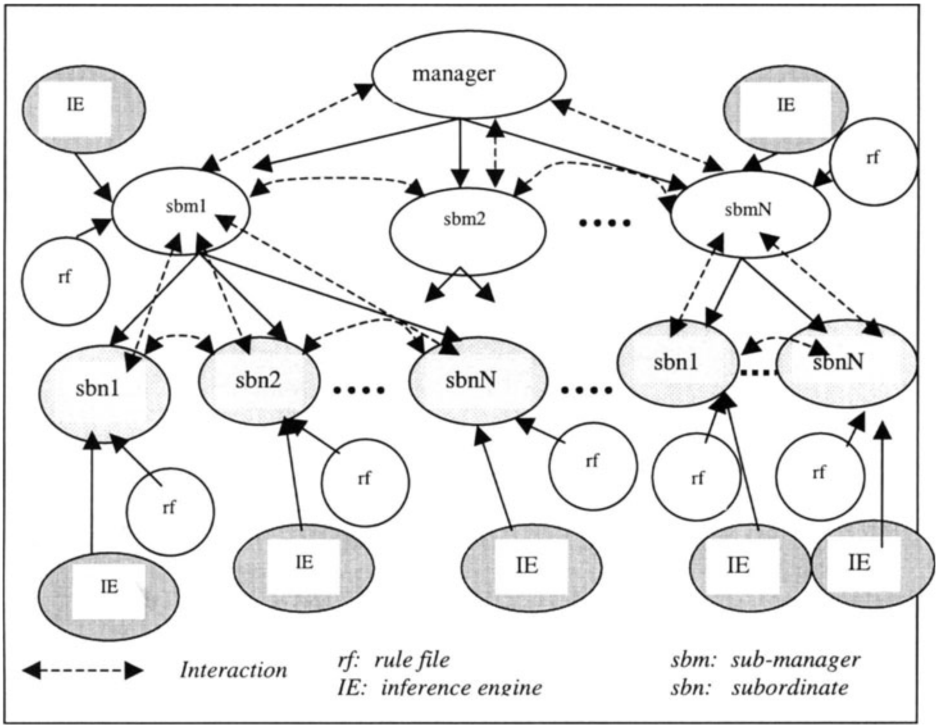


Figure 3: Heuristic Map

5 WATER RESOURCE MANAGEMENT

The HEURISTIC MAP in **Figure 4** depicts a water resource management application, a prototype developed in collaboration with the Geelong Water Board (Australia). It is designed to capture the four requisite knowledge types:

- Domain Knowledge
- Structural Knowledge
- Strategic Knowledge (e.g. Actors)
- Meta-knowledge

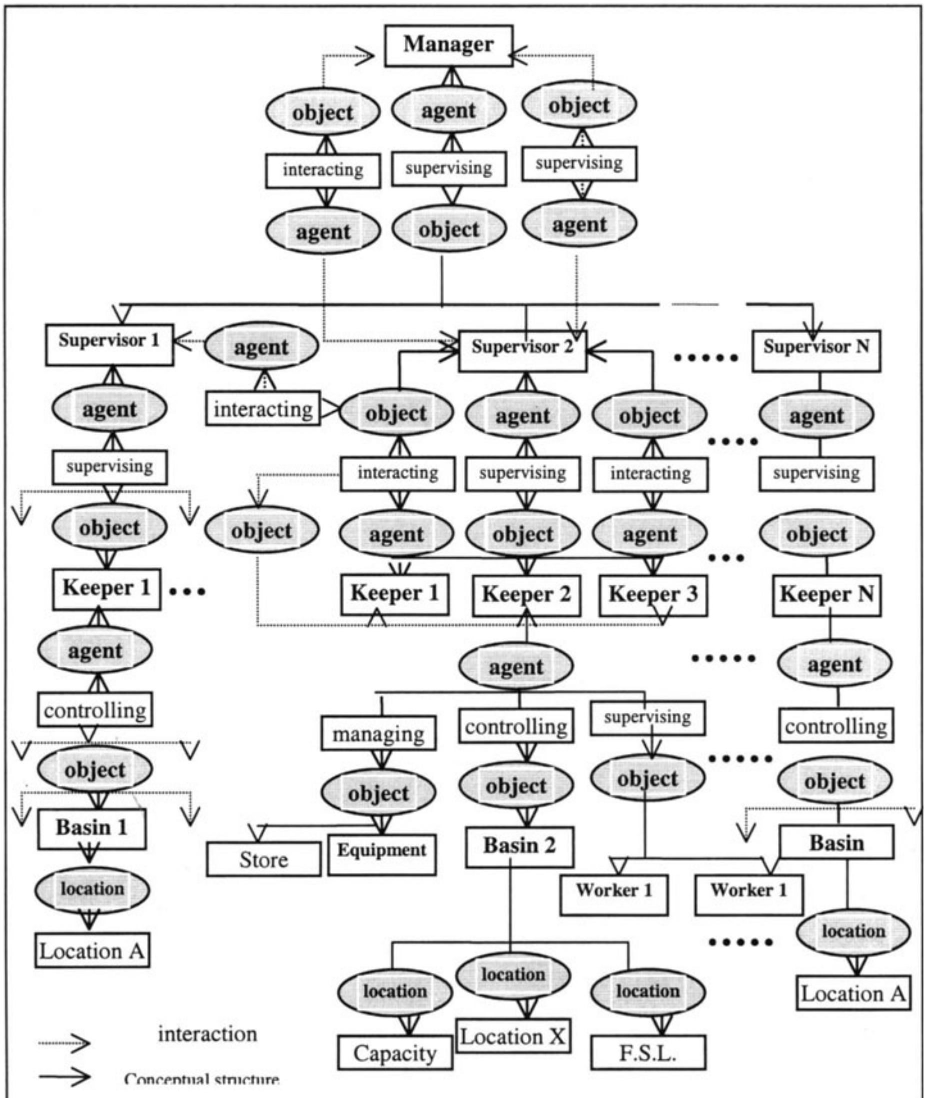
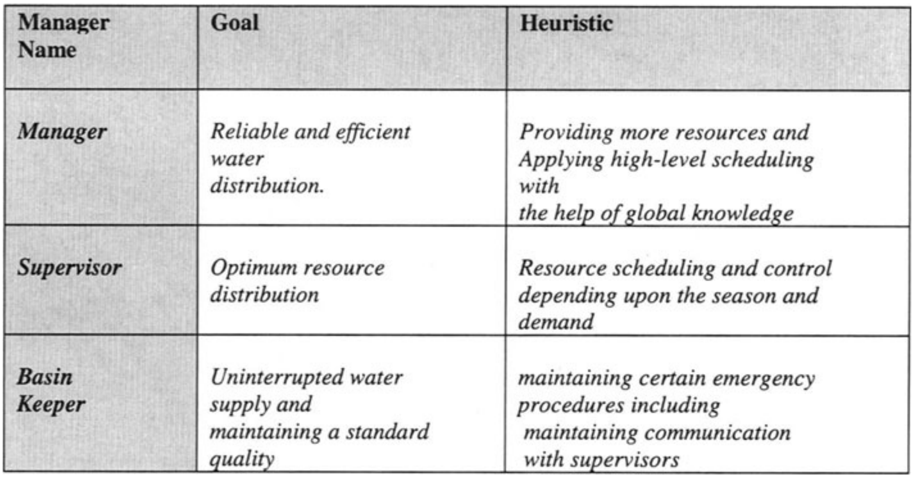


Figure 4: Heuristic Map of Water Resource Management

The goals and heuristics used in the prototypical Water Management System are shown in Table 1.

Table 1: Goals and Heuristic in the Prototype Water Management System



Manager Name	Goal	Heuristic
Manager	*Reliable and efficient water distribution.*	*Providing more resources and Applying high-level scheduling with the help of global knowledge*
Supervisor	*Optimum resource distribution*	*Resource scheduling and control depending upon the season and demand*
Basin Keeper	*Uninterrupted water supply and maintaining a standard quality*	*maintaining certain emergency procedures including maintaining communication with supervisors*

In the management hierarchy the manager, the supervisor and the basin keeper are primarily decision making objects. Each object has a set of heuristics appropriate to the type of decision making. The manager is the highest level object in the management hierarchy and it has the task of looking after several supervisors. The manager object maintains necessary meta-knowledge required for abstract level control and applies one or more from a set of accumulated heuristics when consulted by supervisors.

The supervisor subclass is the second level object in the management hierarchy, and it has the task of managing and guiding the basin keepers. The supervisors consult their manager if a higher level decision is required. It has a set of heuristics to produce decisions when confronted by basin keepers regarding day-to-day running of the water basin. The keeper subclass is the third level object in the management hierarchy. It has the task of carrying out the plant maintenance and water treatment. The keeper has a set of heuristics to make decisions about day-to-day problems occurring at the basin or when the keepers should consult the supervisors for a higher level decision.

Acknowledgements: The authors gratefully acknowledge receipt of a research grant from Deakin University to support this work. We are also grateful to Dickson Lukose and Roland Lim for their technical prototypes (under contract).

6. REFERENCES

1.[Clancey, 85]: *Clancey, W.J., "The Epistemology of a Rule Base System – a Framework for Explanation, "Artificial Intelligence, 27,pp 289-251, 1985.*

2. [Kaemmer, 86] *Kaemmer W.F. and Larson, "A Graph-oriented Knowledge Representation and Unification Technique for Automatically Selecting and Invoking Software Functions, "AAAI-86, 5th National Conference on AI, August, 1986, pp. 824-830.*

3. [Kutti, 92]: *Kutti, S., D. Lukose and R. Lim "Heuristic Map for Resource Management," Technical Report 4/92, School of Computing and Mathematics, Deakin University, 1992.*

4. [Kutti, 90a]: *Kutti, S. and B.J. Garner, "Evaluation of Knowledge-Based Operating Systems Kernel, " Proceeding of the 13th Australian Computer Science Conference, Monash University, Clayton, Victoria, Australia, January 1990.*

5. [Kutti, 90b]: *Kutti, S., and B.J. Garner, Semantic Network as the Basis for Dynamic Object-Oriented Structures, Proceedings of the PRICAI, 90 Nagoya, Japan, 13-17 November 1990.*

6. [Meer, 83] *Meersman R., and F. Van Assche, "Modeling and Manipulating Production Data Bases in Terms of Semantic Nets, "Proceedings of International Joint Conference on AI (IJCAI), 1983, pp 325-329.*

7. [Quillion, 68]: *Quillion M., "Semantic Memory,' in Semantic Inf. Processing, ed: Minsky. M, MIT press, Cambridge, Massachusetts, 1968, pp 227-270.*

8. [Schube, 75]: *Schubert, L.K. "Extending the Expressive Power of Semantic Networks," IJCAI-75, 3-8 September 1975, TBILISI, Georgia, USSR.*

9. [Sowa, 84]: *Sowa, J.F., "Conceptual Structures: Information Processing in Mind and Machine, published by Addison Wesley, Reading, Mass, 1984.*

10. [Turban, 86]: *Turban, E and M Sepehri, "Applications of Decision Support and Expert Systems in Flexible Manufacturing Systems," Journal of Operations Management, Vol.6, No.4, August 1986, pp.433-448.*

11. [Garner, 96]: *Special Issue on Knowledge Acquisition, Representation and Processing KARP'95; Knowledge Based Systems, Vol.9 , No.8, December 1996.*

12. [Garner, 91] *Garner B.J. Forrester, C.L., Lukose, D and Smith, N., (1991) Protoypical Knowledge for Creativity Research, in Proceedings of the Workshop on Creativity and A.I., Queensland, Australia, 1991.*

New Modeling Method for Using Large Knowledge Bases

Wataru Ikeda, Setsuo Ohsuga

Department of Information and Computer Science
School of Science and Engineering
Waseda University
{wataru, ohsuga}@ohsuga.info.waseda.ac,jp

Abstract. The objective of this paper is to discuss some aspects of large scale knowledge bases, especially the ways of building knowledge bases from the view point of using knowledge bases for solving real problems and for assuring knowledge based systems generality. A new modeling scheme for representing problems including strategic decision is discussed as well as a way of generating problem specific problem solving systems. Many multi-level concepts such as a multi-level function structure and its corresponding knowledge structure, multiple meta-level operations, a multi-strata model to represent problems including human activity, etc. It is shown that the system realizes not only the generality but also practicality of problem solving by enabling automatic programming.

1. Introduction

The largest difference between persons and computers in problem solving is autonomy and generality to include the wide range of activities in the actual scope. Here the term problem solving is used in a very wide sense to mean every activity to derive answer meeting given conditions in a given environment. Generality requires a system to process the wide class of problems. Large Knowledge Bases are indispensable for actual intelligent systems from the viewpoint of generality. The objective of this paper is to discuss some issues on large scale knowledge bases and as well the way of building and using them. The approach taken in this paper is to introduce a new multi-level concept so as to enable the system to make decisions dynamically on the way of problem solving [1]. In a sense it is to include many virtual persons in the system [2]. In this paper, various concepts and methods to meet the conditions presented are discussed in Chapter 2. The outline of the concrete representation of knowledge to realize these concepts and methods are in Chapter 3. The condition of large scale knowledge base is discussed in Chapter 4. Chapter 5 is the conclusion.

2. A way of making knowledge based systems general

The condition of a problem solving system being general is that it should be able to deal with different problems in the same system [3]. There are various types of problems such as analysis, design, control, decision making, diagnosis, planning, scheduling, teaching and so on. Most of them are not well dealt with by conventional software method but require the system a capability to find a solution itself in a large space. Since the space is open, self-controlled exploration in the space is necessary. The system must be provided with the various methods to solve the different type of problems, each of which is represented by a specific knowledge chunk. Furthermore, a complex problem concerns different problem domains and since a problem requires domain specific knowledge, the system must be provided with a global knowledge base including the various domain knowledge.

In order to use knowledge effectively, the system must be able to extract only the necessary knowledge from the knowledge base referring to the type and the domain of the problem to be solved. For this purpose knowledge must be well structured.

2.1. Exploratory problem solving

People use a general method of problem solving known as the scientific method today. In this method a problem is represented explicitly, is analyzed and evaluated whether it satisfies the given requirements, and if not, it is modified. This analysis/modification process is repeated until the goal is reached. Representing this method explicitly and giving it to computer systems is necessary for achieving the goal mentioned above. Knowledge is used for promoting this process. A hierarchical object model representation is necessary.

2.2. Meta-operation for defining and controlling problem solving

The search space to be explored is, in the usual case, vast and unforeseen. In the repetitive problem solving scheme of Figure 1, only a small portion of the neighborhood of the current model in the solution space is explored. Guiding the selection of the next model as a candidate is important. Concretely, this is to select a proper model modification operation at object-level. This is a meta-level operation. The selected knowledge being applied to an object model changes the state of the model. In order to give the system a capability of adapting to the changing environment and also the freedom of adding, deleting or changing knowledge, the meta-level operation should also be knowledge-based. Thus the concept of meta-level knowledge strongly concerns the design of knowledge representation language.

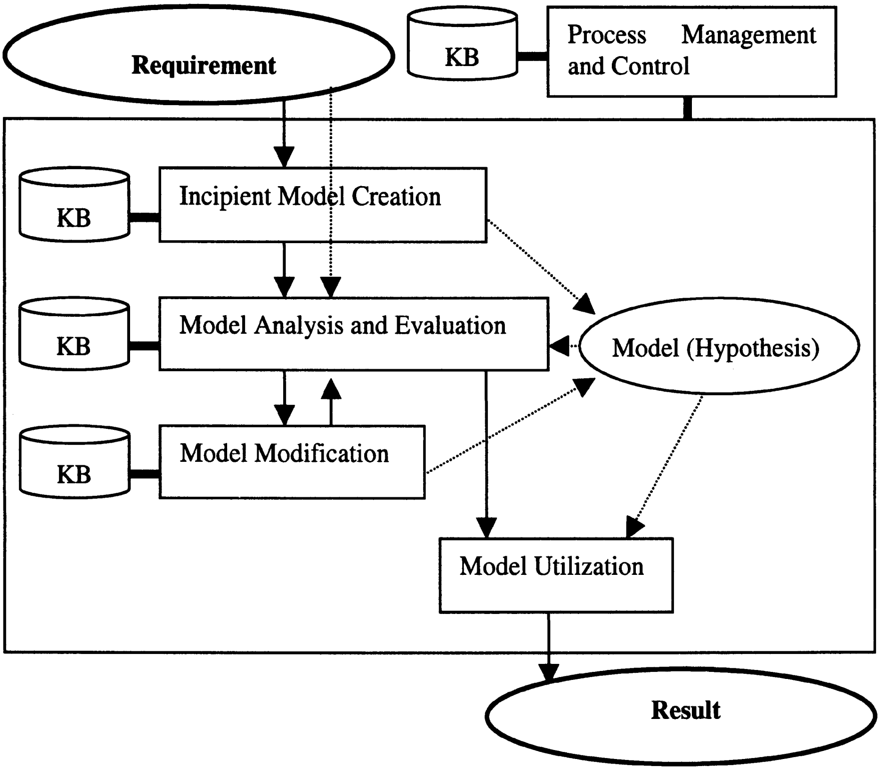


Figure 1: Exploratory problem solving

3. Internal representation of basic concepts

3.1. Multi-strata model as problem model

Building only an object model however is not adequate for representing one's idea and does not initiate any problem solving. A problem arises when some subject has an intention to do something with an object. Every object has many aspects, but only some of them which the subject has interest in and intends to deal with defines the subject's own problem and are included in the model. Different object models may be created depending on personal interests even if the object is the same [4]. A multi-strata modeling scheme to include human intention is necessary as shown in Figure 2.

In this modeling scheme, a base stratum arises from some lowest stratum object ***O1***. This is the basic object model. In the next stratum this problem solving itself is an object ***O2*** and another subject ***S2*** solves the problem created in relation with the object. The requirement given to ***S1*** defines the task the person should do. The upper stratum requirement includes very often the lower stratum one. In order to represent it, meta-level representation scheme is necessary.

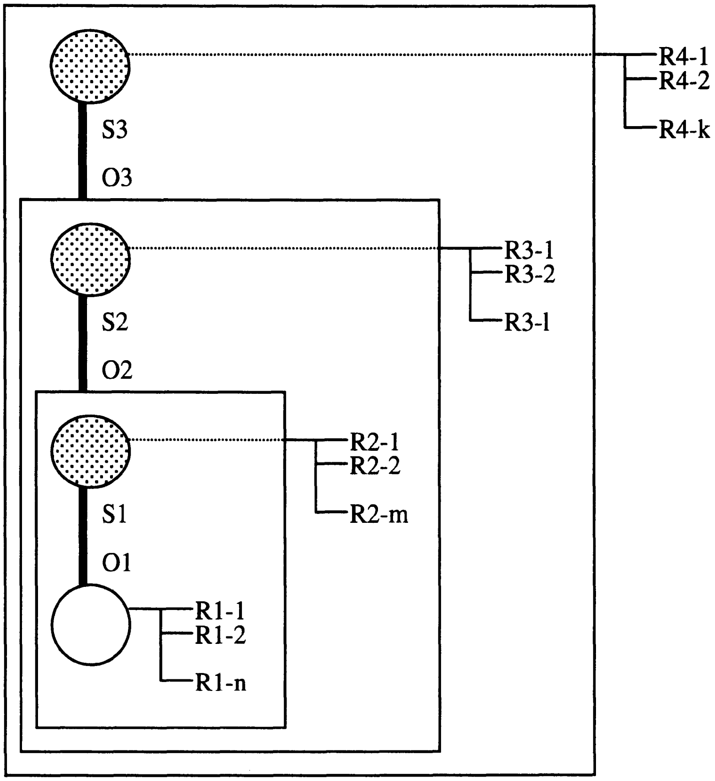


Figure 2: Multi-strata modeling

3.2. Knowledge representation language

A language for representing the concepts discussed so far is necessary. Such a language has been developed with its processor by author's group and named Multi-Layer Logic (MLL) [5]. This language is an extension of first order logic. The main extensions are

1. Introduction of data structure and a method of manipulating it.
2. High order predicate under certain restrictions.
3. Open architecture to accept any procedurally defined predicate.
4. Their combination.

The basic form to represent knowledge is

$$(Q_x x / X)(Q_y y / Y)\cdots(Q_z z / Z)[R(x, y, \cdots, z):-$$
$$P_1(x, y, \cdots, z) * P_2(x, y, \cdots, z) * \cdots * P_n(x, y, \cdots, z)]$$

in which Q_x etc. denotes either the universal quantifier $(\forall)$ or the existential quantifier $(\exists)$, x / X etc. means x in a set X, $*$ denotes a logical connector, i.e., either conjunction or disjunction. This expression is read as "for all/some Q_x x in X, all/some Q_y y in Y, ... , all/some Q_z z in Z, if the relation P_1 and/or P_2 and/or, ... , P_n hold, then R". The prefix is often abbreviated for the sake of simplicity in the followings. An evaluation of predicate is a mapping from a set of variables $(x, y, \cdots, z)$ to either true or false (T,F). This mapping can be performed either by matching with the existing predicate or by a procedure. A predicate to which a procedure for evaluation is provided is named a Procedural Type Predicate (PTP). The others are Normal Type Predicate (NTP). When a PTP is to be evaluated the corresponding procedure is evoked and executed.

Using MLL as a knowledge representation language a knowledge based system named KAUS (Knowledge Acquisition and Utilization System) has been developed by the author's group and used for many problem [6]. The system is designed to accept any Procedural Type Predicate as a pair of predicate and its evaluation procedure.

Some simplified expression is used in the following. Every term can be a universally quantified variable in a rule with a prefix. In this case it must be in the head. The prefix $(\forall x / X)$ for the variable x in the head however is abbreviated except when it is necessary to represent a special meaning. In this case the variable is denoted by the character string starting with an upper-case character showing its domain name. The character string starting from the lower-case letter shows a constant. But if some terms are used to define a local concept, that is, closed within the single formula, these are not necessary to be included in the head but can appear only in the body. Thus, an expression as the following appears.

$$predicate(X_1, \cdots, X_n):-$$
$$(Q_y / Y)predicate'(X_1, \cdots, X_n, y), \cdots, predicate''(X_1, \cdots, X_n)$$

It is of course possible to translate this expression into the standard form in which every quantifier is put in the prefix. But the above expression is used in the following because of its comprehensiveness. Even the prefix in the body is often abbreviated unless it is to be noted for representing a special concept. In the following a predicate starting with # denotes a PTP. As well, the expressions $A:-B, A:-C$ can be merged to a predicate $A:-B+C$ where $+$ denotes the disjunction.

3.3. System generation based on multi-strata model

Each specific problem requires its own problem solving method represented by a problem oriented structure of functions. Let it be called a problem-specific function structure. Because of its problem dependency, the key information to generate this structure must be included in the problem representation in the form of multi-strata model.

The requirements are distributed in this problem model. These are processed from the highest stratum in order to create the problem-specific function structure. For this purpose a special operation to identify and retrieve the necessary knowledge chunks is performed in the system. For example, let a requirement "*automate the activity of subject1*" be given to a highest level subject in the multi-strata model. Let its internal form be "*automateActivity (subject2, subject1, system)*" where "*subject2*", "*subject1*", and "*system*" denote respectively the subject to which this requirement is given, the subject of which the activity is to be automated, and problem specific system to be generated in order to replace the designated subject ("*subject1*") for the purpose of automating the task. The objective is to retrieve necessary knowledge chunks to define the problem specific system from a knowledge base and to replace the lower stratum subject. Knowledge must be well structured to facilitate these operations.

Special high-level knowledge is provided for performing this retrieval operation. If the requirement to the "*subject1*" is a design problem, a set of functional requirements are given and the "*subject1*“ is asked to find such an object model structure that satisfies these requirements. When the requirement is given, the model is not made but a node to represent the model is created and the requirements are given to the node. The model structure is made starting from this node. In this case the requirement for "*subject2*" is satisfied by generating a system which can do every operation required by "*subject1*" to solve the given problem. As has already been discussed, it needs a general scheme of problem solving as shown in Figure 1. Therefore retrieving a knowledge structure to represent this scheme can satisfy the requirement given to "*subject2*". The following rule can be used to achieve this operation.

```
automateActivity(Subject2,Subject1,System):-

#getSubjectRequirement(Subject1,Requirement1),         *1

generateSystem(Requirement1,System),                   *2

evokeSubject(Subject1,System).                         *3
```

This is a general rule. Let, in above case, "*subject2*" and "*subject1*" be substituted by ***S2*** and ***S1*** respectively. This rule says that a system which replaces the lower stratum subject is obtained by, knowing the lower stratum subject (*1), knowing the requirement given to it (*2), and generating a system to satisfy the requirement (*3). The operations (*2) and (*3) are defined by still the other rules. After the Requirement1 is substituted by the requirement to ***S1*** in the multi-strata model, the predicate "*generateSystem*" is evaluated. In the above example, "*exploration(subject1, model, domain)*" is the requirement given to the lower stratum subject and is substituted into " *Requirement1*". Since the knowledge to be used for solving a problem may be different by the requirement substituted into "*Requirement1*", problem types and problem domain, it is assumed that the different knowledge chunk is provided for every case. It is also assumed that the name of the problem domain is included in the requirement. Finally the lower stratum subject ***S1*** is evoked for operation. This predicate is to start a problem solving for the requirement given to the designated subject using the system assigned thereto (*3). The predicates "*generateSystem*" and "*evokeSubject*" include in the above rule is expanded further bye the following rule.

```
generateSystem(exploration(Subject1,Model,Domain),Syste
m):-

#getObjectRequirement(Model, Requirement'),           *1

problemType(Requirement',Type)                        *2

#makeRetrieveKey(Subject1,Model,Domain,Key,KC),       *3

retrieveKnowledge(Subject1,Domain,Key,KC),            *4

makeSystem(System,KC).                                *5
```

There are the different rules for "*generateSystem*" by the requirements included therein. This is the one in which the requirement to "*subject1*" is "solve a problem". KC is an abbreviation of knowledge chunk. "*Subject1*" and "*Model*" are substituted again by ***S1*** and ***O1*** respectively. The type of problem must be identified from the representation of the requirement on "*object1*". If it is represented in the form of "*functionality(object, value)*", there are three different types depending on which of object, value and functionality is unknown. In the case of the object being unknown this is a synthetic type problem like a design problem. First the object level requirement is obtained (*1). The problem type is obtained by the predicate (*2). Then the necessary knowledge chunks are retrieved by (*3,*4). Different rules are retrieved for the different problem types and domains. A problem specific problem solving system can be generated including these knowledge chunk (*5).

The requirement to **S1** may be given more specifically such as design, diagnosis, control, programming etc. then the problem type is included explicitly in the requirement. The rules for "*generateSystem*" are prepared for them. Then the problem type is obtained straight.

```
evokeSubject(Subject,System):-

#getSubjectRequirement(Subject,Requirement1),         *1

getSolution(Requirement1,System).                     *2
```

Problem solving to satisfy the Requirement1 is executed using the System (*2). The knowledge chunks can be retrieved by the following rule.

```
retrieveKnowledge(Subject1,Domain,Key,KC):-

#getTaskKnowledge(Key,TaskKnowledge),                    *1

#getDomainKnowledge(Domain,DomainKnowledge),             *2

#includeKnowledge(KC,TaskKnowledge,DomainKnowledge). *3
```

The first predicate (*1) takes out the knowledge to define a specified task from the global knowledge base. The information on the knowledge chunk is in Key. Since the scheme of performing tasks is common to many domains, this task knowledge includes knowledge chunks that relate to different domains. The second predicate (*2) selects as well the domain knowledge chunk. Finally, these knowledge chunks are integrated to form the knowledge chunks as required for the given domain.

It would be shown that by providing this kind of rules concerning the multi-strata model, the problem-specific problem-solving systems could be generated.

4. Condition for large knowledge bases

The knowledge base discussed above must be large enough to cover wide application areas. Naturally it is very large and is difficult to be held and maintained by a few users but must be opened because a large part of the knowledge is common to many people. There arises the need for a large common accessible knowledge base. Since these intelligent systems require the properly structured knowledge to achieve the goal of solving complex problems, the Large Knowledge Bases (LKB for short) must be designed to meet this condition.

The key for accessing the knowledge chunk must be created in each intelligent terminal system in which the problem solving discussed before is performed. Every intelligent terminal system keeps a directory of knowledge chunks in LKB. If some knowledge becomes necessary in solving some problem but lacked in intelligent system, a key is created and the request is issued to LKB. The intelligent terminal system must have its own knowledge base and the retrieved knowledge is added to it. The intelligent terminal system must also maintain this directory file up-to-date following to the notice from LKB. In other words, LKB must distribute the information to represent the current state of the knowledge base to all intelligent terminal systems. This is the requirement for the intelligent terminal system to access to LKB.

On the other hand, the requirement for LKB must be cleared. It is expected that there can be a lot of accesses to LKB when the use of intelligent systems becomes every day affairs. Then LKB must be provided with a very efficient method of retrieving knowledge. It cannot be the retrieval by pattern matching because it is too slow but must be retrieval by index as is used in the ordinary information retrieval. The difference is that the knowledge chunks are complexly structured according to the relations between intelligent functions. The relations between them such that some knowledge chunks is retrieved via the other knowledge chunks. The characteristics of LKB are as follows. A management system must be designed to meet these conditions.

1. It accepts the request from the intelligent terminals, retrieves the required knowledge chunks and sends them back efficiently.
2. It maintains the knowledge base up-to-date by accepting new knowledge or replacing the old knowledge with the new and the more smart knowledge.
3. It tries to discover knowledge from the collection of data in cooperation with databases.
4. It controls the traffic of information around LKB
5. It preserves the security of the information and the system.
6. It assures the fine control of authorization of users.

What information can be used as the key? The key is an information to describe characteristics of knowledge. Hence this corresponds to meta-knowledge in intelligent system. When a chunk or a world of knowledge is created in an intelligent system, some meta-knowledge characterizing this world must be given. As has been discussed, meta-knowledge is not dependent on each specific problem but on problem domain. That is, the structure of classifying the knowledge is common to a problem domain. This structure must be used also as the schema of LKB. If an intelligent system finds that some knowledge in a chunk is lacked, it uses the higher level knowledge characterizing the chunk as a ken for retrieving. Thus, even if inference is not used for retrieving information, it may be necessary for maintaining the system. Moreover if LKB is to have the capability above, e.g. discovering knowledge in data, the level of its intelligence must be high enough. Since, however, the objective and role of LKB is different from intelligent terminal systems. A different approach in design is necessary.

5. Conclusion

The problems, which arise in the future in many areas of social activities, are expected to glow very large and complex. They are already coming to and soon exceed the limit of human capability. In fact human capability is limited in many aspects such as, the limit for dealing with the large scale problem, the limit for adapting the required speed in doing things, the limit of controlling the error, the limit of understanding the meaning of what is happening, the limit of understanding the trans-disciplinary problems etc. Thus conventional human centered style of problem solving cannot be held anymore but some alternative method as a break-through, at least a strong supporting method, is needed. Such computers are not based on the conventional technology, but the more intelligent systems become necessary. The minimum requirements for the computers are autonomy, generality, and practicality [7]. In this paper generality is mainly discussed. Some parts discussed in this paper have already been tested. These attempts were to prove the validity of the way of approaching the goal. The author's group is coming to the final stage of achieving the goal of this paper as the research work.

References

[1] S. Ohsuga, Toward Truly Intelligent Information Systems - From Expert Systems to Automatic Programming, (to appear in knowledge based systems), 1998

[2] S. Ohsuga, Multi-Strata Modeling to Automate Problem Solving Including Human Activity, Proc. European-Japanese Seminar on Information Modeling and Knowledge Bases '96, IO Press 1997

[3] F. H. Ross, Artificial Intelligence-What Works and What Doesn't?, AI Magazine Vol.18 No.2, 1997

[4] Toppano, E. Chittaro, L. & Tasso, C. Dimensions of Abstraction and Approximation in the Multimodeling Approach, Proc. Fourth European-Japanese Seminar on Information Modeling and Knowledge Bases, 1994

[5] S. Ohsuga and H. Yamauchi, Multi-Layer Logic - A Predicate Logic Including Data Structure As Knowledge Representation Language, New Generation Computing, 1985

[6] H. Yamauchi, S. Ohsuga, KAUS User's Manual Version 6.49, RCAST, University of Tokyo, 1997

[7] T. Aida, Intelligent System Must Be Able To Make Programs Automatically For Assuring The Practicality, Proc. IEA-98-AIE, 1997

Integration of Formal Concept Analysis in a Knowledge-Based Assistant

Baltasar Fernandez-Manjon, Antonio Navarro, Juan M. Cigarran,
Alfredo Fernandez-Valmayor

Escuela Superior de Informática, Universidad Complutense de Madrid
Avda. Complutense s/n, 28040, Madrid, Spain
bfmanjon@dia.ucm.es

Abstract. To build a knowledge base for an intelligent help system, as Aran, is an essential and difficult phase. In Aran, a knowledge based assistant to help users in the use of UNIX systems, we integrate the traditional manual approach of conceptual information structuring with a complementary one based on Formal Concept Analysis (FCA). FCA allow us to obtain the domain formal concepts (semi) automatically and to organise the information around them.

1 Introduction

As the complexity of software applications increases and computers are being used by all kind of individuals, the provision of on line help in complex software applications is becoming crucial. Knowledge-based assistants (also known as Intelligent Help Systems IHS) have been proposed as a way to improve the usability of complex software applications [12,14]. We consider that IHS can integrate smoothly the learning and working processes providing benefits such as a potential increase in users' motivation [2]. Nevertheless, at present, the cost required to build such systems keeps them mostly in the research domain and out of the practical realm [4]. The organisation and structuring of domain information is a major task when producing IHS. Usually these systems are built around a knowledge base or conceptual network that represents key concepts of the application domain (according to domain knowledge experts) [1].

In this paper, we present a new approach to the automatic structuring of information that simplifies IHS construction when dealing with complex domains. The complementary approach that we propose is based on Formal Concept Analysis (FCA) theory. FCA theory can provide the basis for educational tools that use a conceptual network as a learning tool or as a navigational support (e.g. giving access to a rich multimedia documentation), or even as a method for designing educational applications. FCA has been applied successfully in many data analysis applications and it is especially well suited when it is necessary to deal with a big number of entities (or objects) that can be described using a rich set of properties (or attributes). Using FCA we can automatically classify and structure all the information around the "formal concepts" of the domain (also called conceptual classes or categories), which are natural pairs of objects and attributes sets. The FCA approach has been used in the construction of a help tool for the UNIX operating system called Aran. In Aran we integrate as complementary both approaches, the FCA and the more traditional approach based on a static, hand coded, conceptual network used to organise and to structure all the available domain information.

The rest of the paper is organised in the following way. First, we present the basics of the formal concept analysis theory. Then we give an overview of the integration of technologies in Aran, and details on how information is indexed using both methods. Also, we introduce briefly FCA as a tool to assist in the domain analysis. Finally, we present the conclusions of our work.

2 Formal Concept Analysis

Formal Concept Analysis (FCA) is a relatively new approach to the mathematics normalisation and representation of conceptual knowledge [13]. It is a theory of concept formation derived from lattice and ordered set theory that provides a theoretical model for the analysis of conceptual hierarchies. From the Computer Science point of view, it is an automatic technique for information structuring and classification that permits the construction of applications for data and domain analysis.

2.1 Contexts and Concepts

The key idea in FCA is the notion of formal concept around which the data will be structured. Formal concepts are formal abstractions of concepts of human thought that allow a meaningful and comprehensible interpretation of the data. In FCA a concept is determined by its intention (comprehension) and its extension. The extension covers the objects belonging to the concept, e.g. the computers in a department. The intention comprises the attributes shared by all the objects under consideration, e.g. all the Macintosh in the department are multimedia, they all are computers, they all have network connections, and so on. With respect to a specific concept, if an object belongs to the concept and an attribute is valid for the concept, then the object "has" that attribute. The extension and intention of concepts are connected through the "has" relationship between objects and attributes, and clearly are reciprocally dependent.

Because a concept can have many instances, and the set of all instances an almost limitless set of shared attributes (properties) it is customary to work with a specific context in which both the set of objects and attributes are fixed. The mathematical model of the relation between objects and attributes is called the formal context. So, a formal context K is a triple (G,M,I) consisting of two sets G and M and a binary relation I between G and M; the elements g of G are called objects, and the elements m of M are the attributes that objects might have, and gIm asserts that "object g has the attribute m" (or equivalently "the attribute m applies to the object g"). If we are considering the four computers in a small department according to the operating systems, multimedia and network capabilities we have G={C1, C2, C3, C4}, M={pc, mac, multimedia (mm), network (nt)}, I = {C1Ipc, C1Imm, C1Int, C2Imac, C2Imm, C3Imac, C3Imm, C3Int, C4Ipc, C4Int} that we represent in a tabular way:

	pc	*mac*	*mm*	*nt*
C1	x		x	x
C2		x	x	
C3		x	x	x
C4	x			x

Table 1. Characteristics of the four computers

Let A be a subset of the object set G -i.e. A consists only of objects and all these objects belong to G-, then A' denotes the set of all attributes from the attribute set M, common to all the objects belonging to A (e.g. if A = {C1, C4}, A' = {pc, nt}). Conversely, for a subset B of the attribute set M, B' denotes the set of all objects from G, to which each attribute from B applies to, i.e.:

$$A' = \{ m \in M \mid (\forall g \in A)\, gIm \} \qquad (1)$$

$$B' = \{ g \in G \mid (\forall m \in B)\, gIm \}$$

Then the pair (A,B) formed from these two sets is called a formal concept of the context K, if and only if A'=B and B'=A are true - if B consists of precisely those attributes from M which apply to all objects from A, and if equivalently A consists of precisely those objects from G which have all the attributes from B, i.e., if A = {C1, C2, C3}, A' = {mm} and ({C1, C2, C3}, {mm}) is a concept (conversely if B = {mm}, B' = {C1, C2, C3} and (B', B) is the same concept).

2.2 Generalisation-Specialisation Relation

The concepts of a given context are naturally ordered by the generalisation-specialisation relation (subconcept-superconcept) producing a hierarchy for the context K. At the top will be the more general concepts that have a smaller intention and larger extension than any of the more specialised concepts below. Formally, if (A1, B1) and (A2, B2) are concepts of the context K, then (A1, B1) is a subconcept of (A2, B2) if (and only if), A1 is a subset of A2 (or equivalently, B2 is a subset of B1), i.e.:

$$(A1, B1) \leq (A2,B2) \Leftrightarrow A1 \subseteq A2\ (\Leftrightarrow B2 \subseteq B1) \qquad (2)$$

In the previous example the concept ({C1}, {mm, pc, nt}) is a subconcept of ({C1, C2, C3}, {mm}), because the computer C1 is a multimedia pc with network capabilities, that is, we can state more things about the computer C1 than about the group of computers {C1, C2, C3} (we say that the first concept is more specific than the second).

2.3 Fundamental Theorem

Formal Concept Fundamental Theorem states (among other things) that formal concepts with the generalisation-specialisation ordering form a conceptual hierarchy for the context K that is a complete lattice, denoted by B(G,M,I). Concepts are placed in a lattice structure in which meet and join of any combination of elements are given by:

$$\bigwedge_{j\in J}(Aj,\, Bj) = (\cap_{j\in J}Aj,\, (\cup_{j\in J}Bj)'') \qquad (3)$$

$$\bigvee_{j\in J}(Aj,\, Bj) = ((\cup_{j\in J}Aj)'',\, \cap_{j\in J}Bj)$$

This concept lattice not only contains concepts corresponding to each object but also concepts corresponding to the meet and join of other concepts. This feature leads us to solve (in part) the problem that arises when working with automatic knowledge extraction for knowledge based systems. In fact using FCA "richness of knowledge in data stored leads us to richness in knowledge extracted and classified". FCA has the advantage that much of the mathematics required for data manipulation in the applications can be borrowed directly from lattice theory.

Moreover, concept lattices can be represented graphically by line (or Hasse) diagrams. These structures are composed of nodes and links. Each node represents a concept with its associated description (i.e. the intension and the extension of the concept). The links connecting a node to its children specify an "is-a" or subset relation, indicating that the parent's extension is a superset of each child's extension. More abstract or general nodes occur higher in the hierarchy, whereas more specific ones occur at lower level (Fig. 1 shows the lattice of the computer example). That means that most of the FCA applications can be supported by graphical representations that simplify the presentation of information and the interaction with users [7].

3 Integration of Technologies in Aran

Aran is an intelligent help assistant for the UNIX operating system. Aran integrates different "standard" technologies to provide the help facilities. The aim is to simplify user access, selection and understanding of the information needed to overcome the user's current problem, while, at the same time, offering the user the possibility to expand his knowledge of the operating system. The integrated technologies are: hypertext, statistical information retrieval, formal concept analysis (FCA), explicit knowledge representation and user modelling. Aran uses hypertext techniques as a tool to navigate the information and to interact with the system. The information retrieval technique facilitates the access to documents. FCA facilitates the organisation, the knowledge extraction and the access to the information contained in the Unix documentation. The domain knowledge representation simplifies access, understanding and integration of the information for the user. Adapting the system to the individual user's characteristics is done with the information stored in the user model.

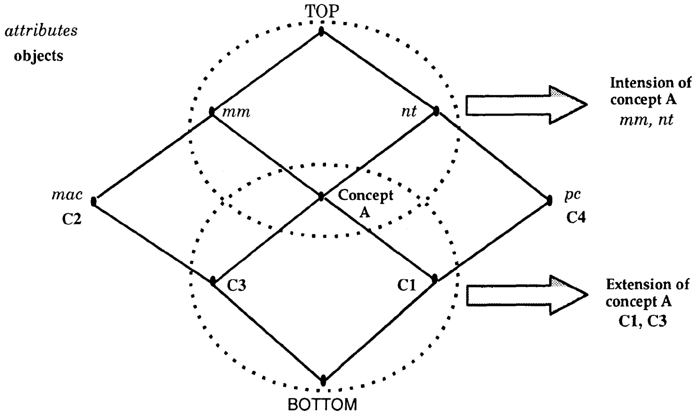


Fig. 1. Lattice of the computer context presented in Table 1

Others help systems present only *ad hoc* information to the user, but Aran reuses the complete documentation that is shipped, in electronic format (manual pages), within the operating system. Three types of indexing is done to this documentation: a) knowledge-based indexing, the documents are indexed using the concepts (mainly the actions) of the domain model; b) statistical free text indexing, the documents are indexed by terms automatically extracted from the text [10]; c) FCA or automatic indexing. In our domain the documents (objects) are related with a set of descriptors (or attributes). This object-attribute relation is used as a formal context to derive all the formal concepts that will be indexed following their generalisation-specialisation hierarchical structure.

Aran provides a direct manipulation, graphical user interface that supports three different, but related, interaction modes. These three operating modes correspond to the three kinds of textual information indexing. In the *browsing mode*, menus and mouse-sensitive representations of the domain model are employed for accessing the domain information and documentation (see Fig. 2). This direct interaction with the domain model will help the user to acquire a complete and accurate model of the Unix system. The *question mode*, where the user makes requests for information using natural language obtaining a ranked list of relevant documents. The *descriptor selection mode*, where the user chooses the descriptors incrementally from a list (provided by Aran) obtaining all documents where those descriptors appear (see Fig. 3). If the user selects a document using the question mode or the descriptor selection mode and he switches to the browsing mode, the visualisation of the domain will be centred in the concepts that index this document.

In Aran we have a complex domain of discourse: in UNIX there are around forty basics commands and more than four hundred user's commands. UNIX is a complex domain where there is no a general agreement on how to organise and on how to present all the implied concepts to the learner [1]. A more detailed description of Aran and of our approach to the IHS construction can be found elsewhere [2, 5].

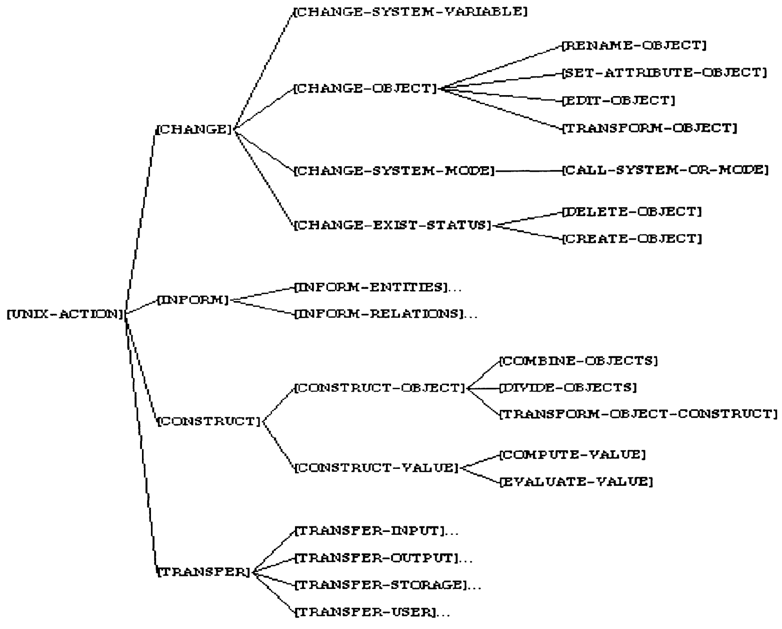


Fig. 2. Partial hierarchy of actions used to index Unix information. The interaction with this domain model is done via a browsing interface.

4 Information Indexing

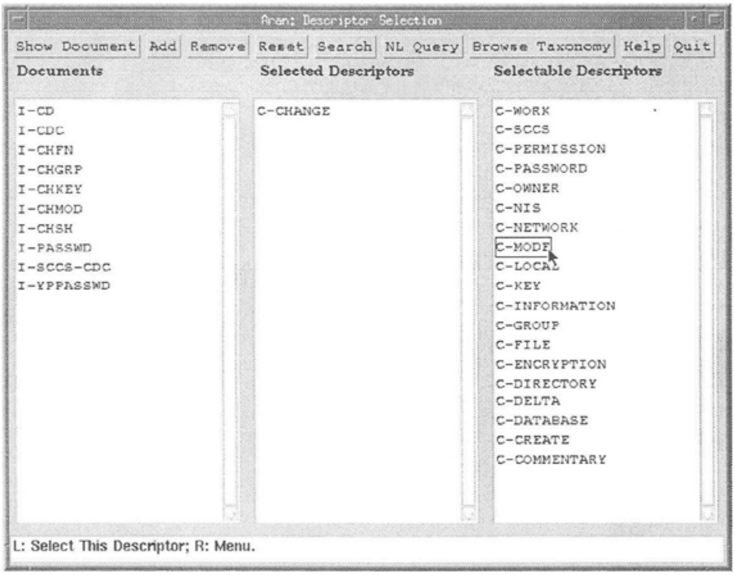
As we are mainly interested in organising and structuring domain information (and not only its access) we will present in more detail the indexing done with the knowledge base and with the formal concept analysis

4.1 Knowledge-Based Indexing

The core of Aran is a knowledge base where the different types of knowledge and information are represented and organised and it includes a model of the domain. The domain knowledge is a conceptual model of the Unix operating system that tries to reflect its information design, technical aspects of the domain as well as a user's view and his/her use of the system. This model allows the organisation and indexing of different kinds of topics around the key concepts of the domain.

We have reused a representation similar to that of the Sinix Consultant [6] obtaining a taxonomic hierarchy of concepts according to different views of the domain concepts. In the knowledge base, the world of Unix concepts is divided into entities, which correspond to Unix (virtual) objects (e.g. file, process, system variable), and actions or operations which involve these objects (e.g. change, communicate with user). Higher-level concepts in the taxonomy reflect more general objects or actions.

We will focus on the taxonomy of actions because command information (and manual pages) is indexed using mainly the actions. Here the formation of general actions and its classification was directed by two independent and orthogonal criteria: thematic and semantic. Actions like system administration or file manipulation are examples of thematic general concepts that group actions with a similar topic or that act on similar objects (correspond roughly with Unix book themes). The basic idea of the semantic classification is to describe the commands by their similar effects (even if they act upon different objects) using a set of "basic" actions that will be later specialised. Change or inform are examples of these kind of general actions.



Documents

I-CD
I-CDC
I-CHFN
I-CHGRP
I-CHKEY
I-CHMOD
I-CHSH
I-PASSWD
I-SCCS-CDC
I-YPPASSWD

Selected Descriptors

C-CHANGE

Selectable Descriptors

C-WORK
C-SCCS
C-PERMISSION
C-PASSWORD
C-OWNER
C-NIS
C-NETWORK
C-MODE
C-LOCAL
C-KEY
C-INFORMATION
C-GROUP
C-FILE
C-ENCRYPTION
C-DIRECTORY
C-DELTA
C-DATABASE
C-CREATE
C-COMMENTARY

L: Select This Descriptor; R: Menu.

Fig. 3. Aran descriptor selection interface. Commands (documents) that contain the descriptor *change* and the list of remaining significant descriptors are shown. Now the user is selecting the descriptor *mode*.

The individual Unix commands and objects are represented and manually indexed as instances of actions and entities, respectively. The commands representation take into account different aspects: syntactic (e.g. name, syntax), semantics (e.g. description text) and pragmatic/tutorial (e.g. related commands, prerequisite concepts, related concepts). From this representation the user has direct access to the related information and documentation.

Loom, a language and environment for knowledge representation and reasoning [9], descendant of the KL-ONE system, is the representation tool that we have chosen to deal with the diversity and complexity of these knowledge sources. Loom has the ability to automatically classify the structured definition of a concept with respect to a taxonomy of other concepts. The knowledge base is implemented as a frame network (with inheritance), where every node represents a concept. Concepts are restricted by a number of slots that relate them to other nodes in the network. Along with this taxonomic-based reasoning, Loom features an object-oriented assertional language which is dynamically truth maintained and supports rule based programming. These characteristics of the Loom systems simplify the construction and management of the knowledge base, in particular the dynamic maintenance of the user model and the general coherence of the domain model [5].

4.2 Formal Concept Analysis Indexing

Aran made a simple use of FCA as a tool to index, according to formal concepts, the initially poorly organised electronic documentation of the UNIX Operating System (manual pages of the section 1, user commands). Normally, documents of the manual have a short description section, with one or two lines, that describes the purpose of the related command. The words used in these descriptions are very significant in the UNIX domain and can be understood by a wide range of users. We have chosen these words, or descriptors, as the attributes, and the commands as the objects of our context. Consequently, in Aran we define a formal context K=(G,M,I), where the objects are the set of commands G, the attributes are the set of all descriptors assigned to these commands, M, and I is the "has" relationship between commands and descriptors. A pair (A, B), where A is a subset of G and B is a subset of M, is a formal concept of this context, if B consist of precisely those descriptors which apply to all commands from A. That way, a subset of commands A will be the extension of a concept, if and only if, there are no more commands that share the same set of descriptors. The equivalent reasoning can be done with attributes.

The set of all descriptors and the descriptors that apply to each UNIX command are semi-automatically obtained from the short description of its related document. First, we use a stop list to remove from the description the common words that are insufficiently specific to represent content -i.e. such as the, of, a, etc. Then words with similar meaning, like plurals or -ing forms, are manually reduced to a single term. This can be regarded as a semi-automatic indexing stage needed to be done initially and once the document collection has changed. This approach is general, domain independent, and can be applied to other collections that do not have a short description [3]. For example, the command "cp" short description is "change work directory" and its assigned descriptors are: "change", "work" and "directory" [11].

FCA indexing simplifies the organisation of the information available and the user access to the documents. Organisation of the documents is made using the automatic clustering feature of FCA. This feature allows the automatic creation of a concept hierarchy to be used in a navigational way and where the user can specialise his information requirements while, at the same time, the system shows him all the possible commands related with his needs. On the other hand, this structure can be potentially exploited by IHS designers to discover complementary ways of organising domain information.

User access to information is implemented in Aran as follows: when the user selects a descriptor (from the list provided by Aran) he accesses the more general concept that includes this descriptor in its intension. Normally, this concept will have subconcepts. The help process is to guide the user from this general concept to its more specific subconcepts. User queries are a set of descriptors that will obtain all commands (documents) with at least those descriptors assigned. For example, if the user selects the descriptor directory (from the list provided by Aran's interface), he will obtain the different commands to manipulate UNIX directories (e.g. cd, dircmp, du, ls, mkdir, organizer, pwd, rm, rmdir) and the Internet user name directory service (e.g. whois). Then all descriptors that apply to these commands are calculated and displayed (minus previously selected descriptors and those shared by all recovered commands). This way the user can select these incrementally compatible descriptors and gain access to more specific subconcepts. Following with the previous example, if the user selects internet, he will only obtain the command whois, and as this command produces a concept object (i.e. it has no subconcepts) the list of selectable descriptor will appear empty. Each step of this interactive process of descriptor selection refines the query and it reduces both the significant remaining descriptors and the selected commands. This process guarantees that no less than one command is obtained and then we can access the related document.

When helping user to access information, Aran does not explicitly or graphically presents to the user domain formal concepts but implicitly uses formal concepts lattice to guide the information search. In Aran we have designed an alternative interface that simplifies the interaction and speeds up the search process (see Fig. 4). With this interface at any step, the user can select any of the more specific descriptors hence it is not necessary to follow, step by step, the lines of the concept diagram (in the Fig. 4 we can see two different search paths in the lattice). This process enables a formally founded and easy to use information access because the induced concept lattice permits fast incremental search with effective feedback to the user [8].

5 Domain Analysis Using Formal Concept Analysis

FCA is also being used in Aran for domain analysis and information structuring. Nevertheless, this part of the FCA module is at this time mainly a designer's tool where we use crude conceptual lattices. As we state in previous section, the UNIX domain is a complex domain that has more than four hundred user's commands, so we start working with subcontexts generated by attributes that we select as specially important. Human computer interaction depends on the size of the lattice and on the information displayed in the lattice. One small lattice could be displayed entirely (not necessarily as a Hasse or line diagram) and medium-large lattices could be simplified showing only the sublattice corresponding to an attribute.

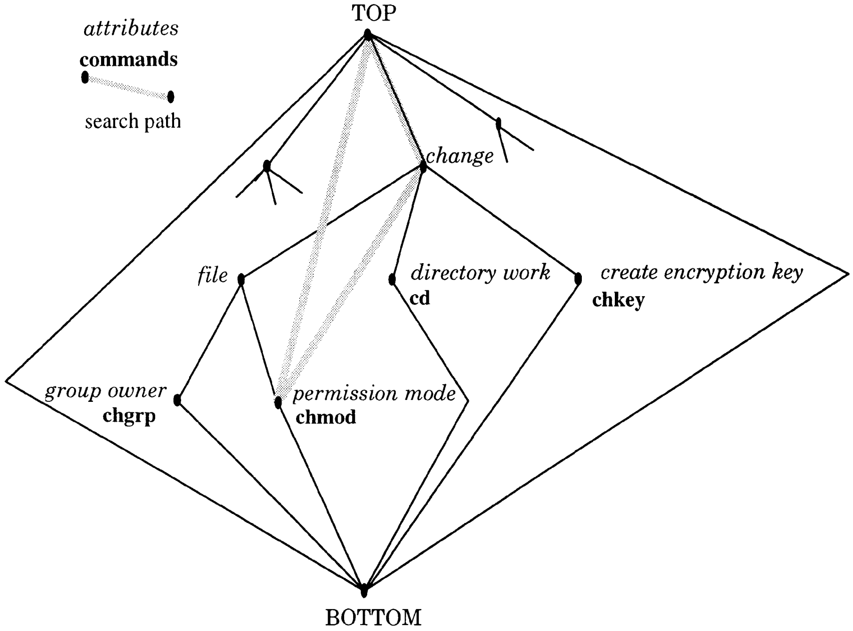


Fig. 4. Concept lattice for several Unix commands with two search paths to access the command *chmod.* One of them correspond to the two steps query specification by the selection of *change* and *mode* -as shown in Fig. 3-. But if the user selects the descriptor *permission* the command is obtained in only one selection.

For example, in Figure 5 we can see the induced lattice of all the commands that have the attribute "job" in its short description. At the first level of the "job" induced lattice we can read out that a UNIX job can be related to “printer”, “queue”, “at”, “remove” and “uucp control inquiry status”, but using our previously stated lattice operations (see Sect. 2.1) we can obtain that {printer}' , {at}' and { uucp control inquiry status}' are disjoint sets of UNIX commands, but that the sets {queue}' and {remove}' have common elements with {printer}' and {at}'. What are the qualitative, or intuitive, implications of these stated results? The first statement tell us that in UNIX we have three different sets of commands: the set {printer}'= {lpr, lpq, lprm} to manipulate printer jobs, the set {at}' = {atq, atrm} to schedule jobs, and the set {uucp, control, inquiry, status}' = {uustat} to report about transfer operations to other systems. The second statement tell us that queue is a common element to scheduled and printer'*s* jobs as are the actions of display and remove elements from the queue. At this point it is important to notice that in our context “at” appears as a “corrupted” attribute, because it is being used as both a time preposition and as the UNIX command "at". In this case it is not a problem because the semantic of the command “at” is clearly related to the meaning of the preposition “at".

Until now we have used this FCA technique as a designer's tool to check Aran's knowledge base, but we are planning to make this information available to users by means of a suitable user interface.

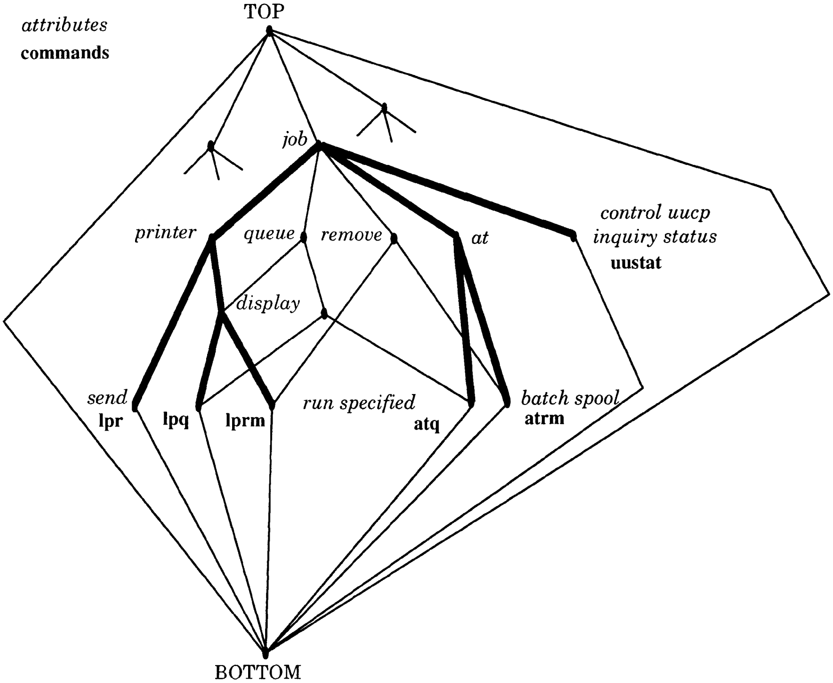


Fig. 5. Concept lattice for the commands containing *job* in its short description.

6. Conclusions

The major contribution of this work is to produce a robust intelligent assistant by means of integrating in the same different techniques. More specifically, we put the stress in the mode how the explicit domain knowledge representation is complemented with formal concept analysis to simplify information access and to enrich the conceptual network. These automatic and manual approach could appear as competing formal tools, where many of the benefits obtained from FCA would be equally obtained using the facilities offered by the LOOM system. But we consider FCA brings many benefits such as that it provides a formal, well founded and easy-to-use approach that can improve the interaction between learner and system. Also, FCA provides a framework in which we can automatically build a conceptual network which organises all domain objects taking into account all its characteristics.

The next steps of this research will be to produce general applications that expand the applicability of FCA as a tool to organise the information handled by other educational software. Also, we are planning to create standalone FCA applications to be used as a tool to simplify domain analysis.

References

[1] Breuker, J. (Eds), 1990. *Developing Intelligent Help Systems.* Report on the P280 ESPRIT Project EUROHELP. EC, Kopenhagen.

[2] Buenaga, M., Fernandez-Manjon, B., Fernandez-Valmayor, A. "Information Overload At The Information Age". Adults in Innovative Learning Situations, Elsevier Science B.V., Amsterdam, pp. 17-30, 1995.

[3] Carpineto, C., Romano, G. "Automatic Construction of Navigable Concept Networks Characterizing Text Databases". M. Gori, G. Soda (Eds.) Topics in Artificial Intelligence, Lectures Notes in Artificial Intelligence 992, Springer-Verlag, Berlin, pp. 67-78, 1995.

[4] Duffy, T.M., Palmer, J., Mehlenbacher B. On Line Help Design and Evaluation. Ablex Publishing Corporation. Norwood, New Jersey, 1992.

[5] Fernandez-Manjon, B., Fernandez-Valmayor, A., Fernãndez Chamizo, C. "Flexibility for Users Needs: Pragmatic User Model Implementation in an Intelligent Help System". British Journal of Educational Technology, **29**(2), 113-123, 1998.

[6] Hecking, M., Kemke, C., Nessen, E., Dengler, D., Gutmann, M., Hector, G. The SINIX Consultant - A Progress Report. Memo Nr. 28. University of Saarlandes, Saarbrücken, Germany, 1988.

[7] Kent, R. E. "Automatic Classification". Intel Corporation surveys, March 1995.

[8] Lindig, C. "Concept-Based Component Retrieval", Proc. IJCAI-95 Workshop on Formal Approaches to the Reuse of Plans, Proofs, and Programs, Montreal, 1995.

[9] MacGregor, R. "The Evolving Technology of Classification-based Knowledge Representation Systems". In: Sowa, J.F. (Ed.) Principles of Semantic Networks, Morgan Kaufmann, CA, 1991.

[10] Salton, G. Automatic Text Processing: the transformation, analysis and retrieval of information by computer, Edit. Addison Wesley, 1989.

[11] Sun. SunOS Reference Manual. Sun Microsystems, Inc. SunOS 4.1.X, 1990.

[12] Wasson B and Akselsen S. "An Overview of On-line Assistance: from On-line Documentation to Intelligent Help and Training". The Knowledge Engineering Review 7 (4) 289-322, 1992.

[13] Wille, R. "Concept Lattices and Conceptual Knowledge Systems". Computers and Mathematics with Applications, 23(6-9), 493-515, 1992.

[14] Winkels R. Explorations in Intelligent Tutoring and Help IOS Press, Amsterdam, 1992.

Knowledge Modeling of Program Supervision Task

Mar Marcos[1]*, Sabine Moisan[2], and Angel P. del Pobil[1]

[1] Universitat Jaume I, Dept. of Computer Science
Campus de Penyeta Roja, E - 12071 Castellón, Spain
[2] INRIA – Sophia Antipolis
BP 93, F - 06902 Sophia Antipolis Cedex, France

Abstract. This paper presents a knowledge-level analysis of the task *program supervision* based on two different systems: PEGASE and PULSAR. A knowledge-level analysis of a knowledge-based system reveals the organisation of the knowledge it uses, and how it uses this knowledge to solve the task. It is also the key to determine the set of properties that it assumes on domain knowledge. These aspects have been successfully used as a framework to compare different systems, mostly for knowledge engineering purposes. Our purpose is to use assumptions as the properties that the knowledge base must verify.

1 Introduction

Program Supervision (PS) consists in the automation of the use of an existing program library, independently of any individual application. Given a user's request expressing a processing goal, a combination of programs has to be selected, scheduled, executed, and the execution monitored to ensure that no unexpected results are produced.

Several systems have been used to supervise image processing libraries for different applications: visual inspection for flaw detection in metal components (VSDE [1]), road-scene obstacle detection (OCAPI [8]), and analysis of interplanetary images (MVP [4]). PS systems are often knowledge-based systems which encapsulate the knowledge about the correct use of programs.

This paper presents a knowledge modeling of two PS systems: PEGASE [10] and PULSAR [9]. Our first aim was to obtain a precise characterisation of these systems in terms of the properties that they assume on domain knowledge, with the purpose of exploiting them in the construction of knowledge base verification tools [6]. A knowledge-level analysis fits well our needs.

A knowledge-level analysis of a knowledge-based system abstracts from implementation details and reveals the organisation of the knowledge it uses, and how it uses this knowledge to solve the task, i.e. which *problem-solving method* it employs [7]. It is also the key to determine the set of properties that the

* Her work has been carried out at INRIA and has been financed by the grant PF95 19899400 from the Spanish Ministry of Education and Culture.

problem-solving method assumes on domain knowledge. These properties make explicit the connection between the problem-solving method and the domain knowledge [2]. They have been successfully exploited for knowledge engineering purposes, e.g. to find out to which domains a certain problem-solving method is applicable, or which problem-solving method is more suitable for a given domain [2]. Our purpose is to use the assumptions of the problem-solving method over the domain knowledge as the properties that the knowledge base must verify. A knowledge-level analysis, however, has other interesting outcomes.

This paper is organised as follows. First the CommonKADS-based framework for our knowledge modeling is described in section 2. Section 3 briefly introduces PS systems. The task model is presented in sections 4, 5, and 6. Finally, section 7 presents the conclusions.

2 A CommonKADS-based framework

Different frameworks for the specification of knowledge-based systems follow the CommonKADS model of expertise [11]. In CommonKADS the following elements are used to describe a knowledge-based system:

- *Task definition*, which describes the problem that the system solves as an input-output relation.
- *Problem-solving method* (PSM), which describes how the task is achieved, including the main reasoning steps and the control over them.
- *Domain ontology* and *domain model*. A domain ontology refers to the vocabulary used to express the domain knowledge. A domain model is the domain knowledge structured as the PSM requires for reasoning.

The framework used in [2] includes the features that the PSM requires on domain knowledge (*assumptions* of the PSM). In [5] other types of assumptions are considered: the assumptions of the task over the domain knowledge, and the assumptions needed to ensure that the PSM can solve the task. Figure 1 shows how assumptions relate the elements of a knowledge-based system.

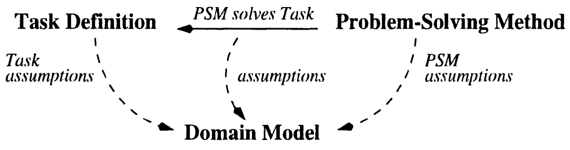


Fig. 1. The elements of a knowledge-based system and their assumptions

In this paper we deal with the concepts of task, PSMs, domain model, and PSM assumptions. In the next section we introduce the architecture of the PS systems that have been the object of our knowledge modeling. Afterwards we present PS domain model, PS task and PSMs (PS methods henceforth), and the requirements or assumptions of PS methods.

3 Program Supervision Systems

PS systems are knowledge-based systems which encapsulate the knowledge about the correct use of a library of programs. A PS system is composed of a reusable PS engine, a knowledge base capturing the expert's knowledge about the use of programs, and the library itself. Next we briefly describe these parts.

3.1 Knowledge base

The knowledge base encapsulates expertise on programs and processing. It contains instances of PS specific concepts such as processing goals, operators (corresponding to programs and their use), operator arguments and parameters, etc. The knowledge base may also include expertise to perform automatically different actions, such as initialisation of operator parameters, evaluation of the results of operator execution, etc.

3.2 Engine

The behavior of a PS engine can coarsely be divided into four steps. First a planning step determines the best (partial) plan to reach the goals defined by the user's request. Then the execution of the (partial) plan is triggered, i.e. the individual programs in the plan are executed. The results of the program execution are passed on to an evaluation step that assesses their quality. This evaluation can be done either automatically by using the expertise in the knowledge base or interactively by the user. Finally if the assessment of results is negative, a repair step applies the appropriate corrective action. Otherwise the process continues with the planning step for the remaining (sub)goals, if there are any.

Although the described behavior is quite general, variations are possible. For instance, at high level, planning and execution may be interleaved because each planning step may depend on information that is only available after the execution of previous programs in the plan. At a lower level, some basic steps can be performed in a more or less complex way.

The knowledge analysis presented in this paper is based on two PS engines: PEGASE and PULSAR. Both can be seen as successors of OCAPI [8] which improve it in different manners. OCAPI performs planning using hierarchical plans and only provides operator re-execution as repair mechanism. PEGASE [10] incorporates richer repair mechanisms. PULSAR (in concrete, PULSAR-IU in [9]) uses a combination of operator-based and hierarchical planning, and a subset of PEGASE repair mechanisms.

4 Program Supervision Domain Model

In this section we present PS concepts structured as PS methods require, i.e. the PS domain model. The concepts taking part in PS are the basic ones we have mentioned above, i.e. goal, operator, argument, and parameter, but also those needed to describe PS task, such as operator knowledge base, problem specification, and problem solution.

4.1 Basic Concepts

Goal A goal represents a processing function that a PS system can perform. It is often used to establish the link between a function and the operators that achieve it.

Operator An operator describes either an individual library program or a more or less complex combination of programs. Two types of operators are distinguished. A *primitive operator* describes a library program. A *compound operator* is described by means of a set of suboperators (or decomposition), which can be in turn compound or primitive ones. The decomposition of a compound operator can represent, e.g. a specialisation (choice), or a sequence. By means of decompositions operators are described at different levels of abstraction. The description of an operator at all levels of abstraction constitutes a *hierarchical operator.*

An operator contains two types of information, namely about the function it performs and about the resolution process it employs. Regarding operator function, we find the information to allow operator selection in the appropriate situations:

- functionality or function that the operator performs.
- characteristics or typical properties of the operator.
- input and output arguments.
- parameters, i.e. tunable arguments.
- preconditions or logical formula stating the applicability conditions of the operator.
- effects or logical formula representing the operator side-effects.
- postconditions or logical formula that must hold after execution.

Regarding operator resolution process, the information depends on the type of operator. The information common to both types of operators is:

- initialisation criteria to initialise parameter values.
- evaluation criteria to check the results of operator execution and detect and diagnose any problem. The diagnosis is expressed as a global assessment on the operator application and/or tuning combined with an assessment on an argument or parameter.
- repair criteria indicate how diagnosed problems have to be solved. The alternatives can be re-executing the operator, transmitting the problem to another operator, or backtracking to a choice point. In case of operator re-execution, adjustment criteria are needed for parameter adjustment.

Primitive operators include information regarding the calling interface of the library program (program call). Compound operators include:

- data distribution or the way in which input (output) arguments or parameters are connected to the input (output) of their suboperators.
- in sequential decompositions, data flow or the connections among output and input arguments of the suboperators in the sequence.
- in specialisation decompositions, choice criteria allowing to select a candidate for the specialisation in the current situation.

Criteria The different criteria that have been mentioned have typically the structure of rule bases. Other alternatives are possible, however.

4.2 Additional Concepts

Operator knowledge base The operator knowledge base groups a set of operators, compound or primitive ones, and possibly a set of goals describing them at a more abstract level.

Problem specification A problem specification consists of the intended processing goal, the input data, the type required for the output data, the initial state (domain objects constituting the problem context), and the constraints that must hold in the solution state.

Problem solution A problem solution consists of a plan, which is a set of operators aimed at solving the problem specification, plus the final state with the results of the execution of this plan. Alternatives for plan structure are, e.g. ordered set of operators or unordered set of operators.

5 Program Supervision Task and Methods

The *PS task* receives as input a problem specification together with an operator knowledge base, and produces as output a plan, a final state, and a result description indicating if the plan was successfully executed or not.

The analysis of PULSAR and PEGASE has revealed the basic subtasks of PS. Figure 2 depicts the high-level subtasks using a task-method decomposition structure.

We see that supervise can be performed by a method which consists of an initialisation step plus the step plan an execute. PULSAR and PEGASE methods for plan and execute are different, as well as other methods at lower levels. In the following paragraphs we describe these different realizations and elaborate on the assumptions they make on domain knowledge (labelled from *(a)* to *(o)*). Notice that, due to a lack of space, no explicit representation of control is given.

Plan and execute This subtask consists of the steps expand plan, select plan, and execute plan.

PEGASE plan and execute performs one plan expansion and then searches for a plan that, after a successful execution, solves the problem. PEGASE only performs one expansion under the assumption that individual operators contain enough information to solve a problem *(a)*, and therefore it is more suitable for knowledge bases containing hierarchical operators.

PULSAR plan and execute searches for a solution trying every additional plan expansion (recursive call to PULSAR plan and execute) once an expanded plan has been successfully executed. It makes no assumption on the organisation of the operator knowledge base.

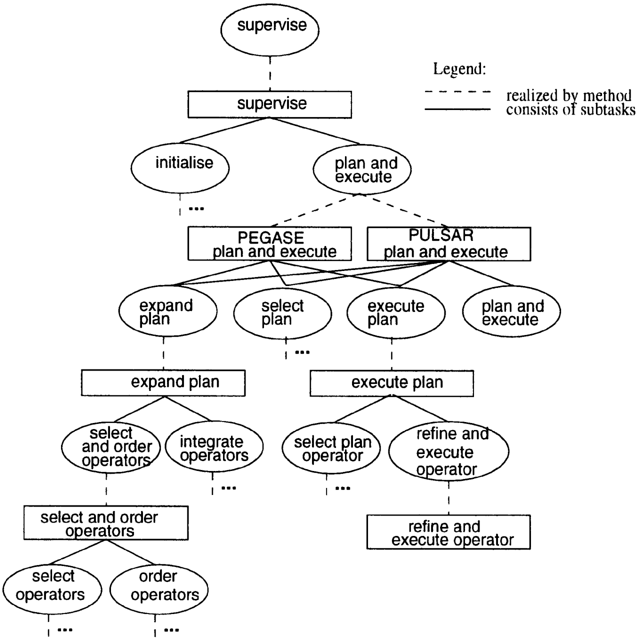


Fig. 2. A task-method decomposition structure for program supervision

Expand plan It consists of select and order operators and integrate operators. According to some heuristics, it selects the candidate operators from the operator knowledge base, orders them, and then integrates them in the current plan. The result is a set of expanded plans. This task is similar in both engines. The differences reside in the heuristics used in its subtasks.

Select and order operators Different heuristics can be used to determine the operators to be incorporated in the current plan and to order them. These differences are made explicit in the alternative realizations of select operators and order operators. The used heuristics impose assumptions *(b)*. For instance, PULSAR select operators is based on a matching between problem specification constraints and operator effects, and thus assumes that the operator knowledge base contains knowledge about effects.

Integrate operators The methods to perform integrate operators depend on the plan structure and hence they make assumptions on this structure *(c)*. In PEGASE a plan is a single (hierarchical) operator whereas in PULSAR it is an unordered set of operators. As PEGASE integrate operators and PULSAR integrate operators simply incorporate the operator without any commitment on the order, no more assumptions are made.

Select plan Once all possible plans have been expanded, one of them must be selected to be executed. This can be done in a more or less smart way, however, the method used in both engines consists simply in selecting the first plan. No assumptions are made.

Execute plan It corresponds to the selection of an applicable operator from the plan (select plan operator), and its refinement or execution (refine and execute operator). This is done in PEGASE merely once, again due to plan structure. PEGASE select plan operator is trivial and makes no assumption for the same reason. PULSAR proceeds along the operators in the plan, while the evaluation of the execution results is positive. PULSAR select plan operator performs selection based on operator preconditions. It assumes that the operator knowledge base contains this knowledge *(d)*.

Refine and execute operator This subtask is showed in figure 3. First the operator parameters are initialised and then the operator is executed if it is a primitive one, specialised or decomposed otherwise. Afterwards the execution results are evaluated, and if necessary the operator is repaired. Each of these actions corresponds to a subtask.

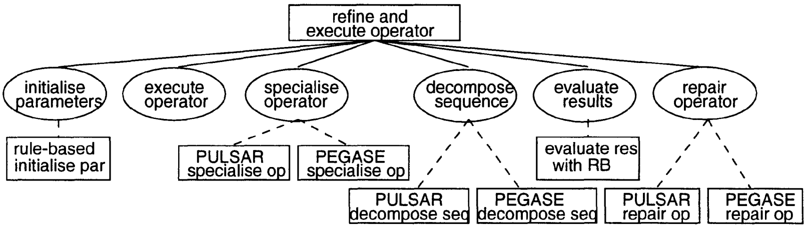


Fig. 3. A task-method decomposition structure for refine and execute operator

Initialise parameters For a correct parameter initialisation, the initialisation criteria of the operator must in principle give means to initialise all parameters *(e)*. This assumption is made concrete in the realization of the subtask. The method used in both engines is rule-based initialise parameters, which consists in the application of forward chaining to the initialisation criteria of the operator. The first assumption is that the initialisation criteria have the form of a rule base *(f)*. It makes other assumptions on the contents of this rule base, e.g. that there is at least one rule for parameter (completeness of the rule base with respect to the parameters set), and that the rules initialising a parameter cover all possible situations (completeness of the rule set initialising a parameter), *(g)* and *(h)*.

Execute operator This subtask actually calls the program in the library and collects the execution results.

Specialise operator This subtask first obtains the candidates for the specialisation in the current situation using the choice criteria of the operator (performed by rule-based choose operators). Then, PEGASE refines and executes the best candidate in terms of the number of times it has been chosen. PULSAR, nevertheless, searches for a suboperator that can be successfully refined and executed in the whole set of candidates. PULSAR assumes that choice criteria can select multiple candidates *(i)*, and thus it is more suitable for choice criteria selecting more than one candidate.

Decompose sequence This subtask obtains the suboperators in the decomposition, and refines or executes every suboperator sequentially. PULSAR performs an intermediary evaluation step after treating each suboperator, as well as the corresponding repair step. It assumes that the operator has knowledge about the required performance of the individual suboperators in the sequence *(j)*.

Evaluate results The method used in both engines for this subtask is rule-based evaluate results, which implies the application of forward chaining to the evaluation criteria of the operator. It assumes that the evaluation criteria have the form of a rule base *(k)*.

Repair operator This subtask first obtains the appropriate repair action in the current situation. This is usually done by rule-based repair operator, which performs one forward chaining step on the repair criteria of the operator and thus assumes that they have the form of a rule base *(l)*. Then either a re-execution (with the corresponding parameter adjustment) or the appropriate repair subtask is activated depending on the repair action. This repair subtask in general propagates the problem to a target operator and forces its repair. PEGASE and PULSAR differ in the repair subtasks that they incorporate.

Since repair operator treats the problem diagnosed by the evaluation step, a basic assumption is the completeness of the repair knowledge with respect to the knowledge used for evaluation *(m)*. This means that all problems diagnosed by the evaluation criteria are treated in the repair criteria. Moreover, the repair knowledge must be complete with respect to the problems propagated to the operator by the repair knowledge of any other operator *(n)*. A different assumption concerns the correctness of repair paths in case of problem propagation *(o)*, i.e. the chaining of repair actions must be continuous and end with an action to solve the problem, e.g. a re-execution.

6 Assumptions of Program Supervision Methods

The assumptions presented have been identified informally from the information on the knowledge utilisation provided by the (sub)task-method decomposition description. Particularly:

- each task determines the precise role of the knowledge it uses and imposes general assumptions.

- the detailed description of the method that performs the task usually imposes additional assumptions. They can make reference to required knowledge, to the structure or form of the knowledge, or to any other requirement so that the method works properly. They can also restate the general assumptions of the task.

For example, initialise parameters assumes that the initialisation criteria give an initial value to all parameters *(e)*. The method rule-based initialise parameters requires that the initialisation criteria have the form of a rule base *(f)*. At the same time, it restates the general assumption taking into account this form, *(g)* and *(h)*.

The conceptual organisation for assumptions in [3] distinguishes epistemological, pragmatic and teleological assumptions. *Epistemological assumptions* refer to the knowledge required by the PSM. These are further divided into *availability* and *property* assumptions, both referring to domain knowledge. In [6] we proposed to distinguish the properties that are critical for the proper functioning of the method (*compulsory*) from those that are only advisable (*desirable*).

In PS methods we find mainly:

- assumptions on availability of knowledge to perform certain tasks (availability assumptions). Examples related to the use of selection/ordering heuristics are *(b)* and *(d)*. A different one is *(j)*.
- assumptions about required knowledge structure (property assumptions). See *(c)* and *(f)*.
- assumptions on other knowledge characteristics, critical or not (compulsory or desirable property assumptions). See *(g)* and *(h)* as examples of compulsory properties, *(a)* and *(i)* as desirable ones.

The assumptions of PS methods specify what they need to operate. They can be exploited to select a PS method for a target domain. For instance, assumption *(a)* of PEGASE plan and execute implicitly states that it handles hierarchical operators. This feature, although too structure-oriented, can be used to select the method for domains in which operator functions are "naturally" seen as combinations of actions [2].

7 Conclusions

The characterisation of PS systems in terms of the assumptions they make on domain knowledge can be used for verification purposes. Actually, the assumptions or requirements of a PS system indicate the properties that the knowledge base should verify [6]. This was the initial motivation of our knowledge modeling.

Furthermore, PS system characterisation can be used to determine the adequacy of the system to the target domain. This is an important issue when deciding which PS system is to be used for engineering a new application, because the success of the final application will often depend on the adequacy of the chosen system to the features of the domain rather than on the efficiency of the algorithms that implement that system [7].

Another interesting result is that the PS task-method decomposition can be seen as a high-level specification of PS methods that could be used to design new PS systems, e.g. by changing the control of a given method or incorporating new steps. Jointly with method assumptions, it can be used to design the knowledge acquisition or verification tool adapted to a given PS system.

Based on method assumptions, we are currently implementing the verification modules necessary to verify knowledge bases for PS systems. We also investigate the use of a software verification tool for the detection of hidden assumptions, in the line of the work in [5].

References

1. R. Bodington, E. Gallier, and C. Theot. The Vision System Development Environment (VSDE): A software environment for the automatic configuration of inspection systems. Draft version, 1992.
2. R. Benjamins, L. Nunes de Barros, and A. Valente. Constructing Planners Through Problem-Solving Methods. In *10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, 1996.
3. R. Benjamins and C. Pierret-Golbreich. Assumptions of problem-solving methods. In N. Shadbolt, K. O'Hara, and G. Schreiber, editors, *Advances in Knowledge Acquisition*, number 1076 in Lecture Notes in Artificial Intelligence. Springer-Verlag, 1996.
4. S. Chien. Automated synthesis of image processing procedures for a large-scale image database. *IEEE International Conference on Image Processing*, pages 796–800, 1994.
5. D. Fensel, A. Schönegge, R. Groenboom, and B. Wielinga. Specification and Verification of Knowledge-Based Systems. In *European Conference on Artificial Intelligence workshop on Validation, Verification and Refinement of Knowledge-Based Systems*, 1996.
6. M. Marcos, S. Moisan, and A. P. del Pobil. A Model-Based Approach to the Verification of Program Supervision Systems. In *4th European Symposium on the Validation and Verification of Knowledge Based Systems*, pages 231–241, June 1997.
7. L. Nunes de Barros, A. Valente, and R. Benjamins. Modeling Planning Tasks. In *3rd International Conference on Artificial Intelligence Planning Systems*, 1996.
8. M. Thonnat, V. Clément, and J. Van den Elst. Supervision of Perception Tasks for Autonomous Systems : The OCAPI Approach. *International Journal of Information Science and Technology*, 3(2):140–163, 1994.
9. J. van den Elst. *Knowledge Modelling for Program Supervision in Image Processing.* PhD thesis, Université de Nice, October 1996.
10. R. Vincent. *Etude des mécanismes de gestion des erreurs dans les systèmes à base de connaisances de pilotage de programmes.* PhD thesis, Université de Nice, March 1997.
11. B. Wielinga, H. Hassan, O. Olsson, K. Orsvärn, G. Schreiber, P. Terpstra, W. van de Welde, and S. Wells. Expertise Model Definition Document. Technical Report KADS-II/M2/UvA/026/5.0, 1994.

Quadri - Dimensional Interpretation of Syllogistic Inferential Processes in Polyvalent Logic, With a View to Structuring Concepts and Assertions for Realizing the Universal Knowledge Basis

Ion I. Mirita
University of Petrosani, str. Universităţii Nr. 20, RO 2675 Petroşani , Romania

Abstract. Modelling syllogistic - inferential processes in polyvalent logic by diachronic syllogistic structures, we realise their QUADRI DIMENSIONAL interpretation, in the paper, by relational - objectual - propertational chains convergent in diachronic spaces. Aristotle considered the definition the motor nerve of syllogistic deduction, the medium term being a definition. Leibnitz conceived the definition as the beginning and end of any demonstration, a demonstration being nothing but a chain of definition. The concept of structure, implying a topological relational approach designates the necessary relations between the elements of a system, invariant and independent of the elements, therefore formalizable the structure constituting an abstract model capable of making the rules, governing the transformations, rationally intelligible. Structuring the concepts and the assertions of scientific theories according to the rules of syllogistic definability and deductibility systems are obtained, which underlie the realization of the Universal Knowledge Basis.

1 Theoretical Basis. Conceptualizations. Formalization.

Generating systems, formalization fulfils the function of thorough analysis of knowledge fields. Concept formalization implies their analysis, contributing to their clarifying and explanation. Formalization facilitates the understanding of demonstration or theory, clarifying and consolidating demonstrations and reasoning.

Concepts can be considered results of formalization or abstraction, serving as instruments of thinking and research which enable us to save brain resources. Scientific theories consist of bodies of concepts and sets of assertions. The problem of understanding a concept or that of verifying the truth of an assertion implies a start from small number of concepts and primitive proposition named axioms or postulates. A concept can be explained or defined by means of other concepts.

The truth of an assertion is to be inferred from other accepted assertions. Starting from small number of ideas and primitive propositions, the lineal approach gives the possibility of concentrating matters of significance and truth in the initial primitive elements; it also involves typical modalities of definition and inference. If the propositions and concepts of a theory are disposed according to definability and inferability links, an axiomatic system of the theory can be obtained.

Aristotle turns definition into the motor of syllogistic inference, the medial term being a definition. According to Leibniz, definition is the beginning and the end of any demonstration, the latter being nothing but a chain of definitions. E. Russel states that "definition is undefinable and it is not even a definite notion".In traditional logic treatises it is shown that a definition is asserted by the genus proximum et differentia specifica. As a restriction, it is pointed out that a definition shouldn't be constructed "idem per idem"; it shouldn't be tautologic as it is impossible to define the definite by means of a more developed form of the false definition "circulus in definendo" or "dialela" each thing should be defined by means of another, either of them being defined by other's elements.

The concept of structure designates the "constellation" of necessary relationships, invariable and independent from the elements, therefore formalizable, which offer the explanation of the "coder of all the possible transformations within the given system. A system becomes completely unintelligible if its parts are studies separately, as it has new properties, distinct from those of its components and not derivable from their sum. By constructing abstract models, it is possible to observe invariable relations which explain the structure and dynamics of systems. J. Piaget defines the structure of a system a coherent assembly of transformations, which ensures the selfregulation of a totality.

By a concept of structure as an abstract model, the rules govern transformations and ensure the functionality of a system become rationally intelligible.In the methodological strategy of structuralism, the rule of diachronic variation enables the explanation of system variations by structural variants. There is a distinction between "synchronic" which designates the relationship between coexistent terms, and diachronic" which refers to the relationship between successive terms. Therefore, structural analysis consists of a topological and relational approach.

Applying structural analysis rules, especially the immanence rule, analysis is exclusively focused on the interior of the investigated field, operating temporarily, from methodological reasons, a closing of the respective field. The interval structure of a knowledge field is established not on grounds of resembling, but of differences, by grouping and ordering differences, more exactly binary opposition, where there are complementary relations between the elements. The activity of ordering differences or binary opposition will be named dichotomic division.

Dichotomic division consists of dividing a field associated to an object into a species object and its complementary, so as the following relations should be observed:

$$O_{00}=O_{ik}\cup O_{ik+1}\quad O_{ik}\cap O_{ik+1}=\varnothing \qquad \textbf{(1)}$$

1.1 Axioms - The Fundamentals Of Aristotelian Syllogistic Construction

Aristotle was the first to formulate ideas on the deductive method of logic. Transposing his ideas in the world of current concepts, by deductive science, Aristotle understands a system S of notions and sentences made up so that:

a) all the sentences in the system S should refer to one and the same domain of objects and relations between objects;

b) any sentences in the system S can be a true sentences;

c) if certain sentences belong to the system S, then other sentences which can be deduced from them according to the laws of logic have to belong to the system S;

d) a finite set of notions should be given in the system S, so that their meaning should not need any explanation, while the meaning of the remaining notions in the system should be defined with the aid of the first group of notions;

e) in the system S a finite number of sentences should be given which are constructed in a such a way that their truth should be evident, while any other sentence in S could be deduced from these sentences according to the laws of logic. The sentences whose truth is evident and which are placed ahead a deductive system are called axioms in traditional logic. The axioms were taken by Aristotle as fundamentals of his syllogistic.

Wang Hao consider that any scientific theory comprises a body of concepts and a large number of assertions.

The meaning of a concept can be explained or defined by means of other concepts.

The truth of an assertion can be determined by deducing it from other accepted assertions. When the concepts and the sentences of a theory are arranged according to the definability and deductibility relations, an axiomatic system of theory is obtained.

1.2 On Structure And Structural Analysis Rules

The structure concepts designates "the constellation"of necessary invariant relations, independent of the elements, therefore formalizable, which offer the "code" explanation to all the possible transformations within the given system.

By constructing abstract models, invariant relations are detected, which can explain the system physiognomy and dynamics.

J.Piaget considers the structure of a system as an ensemble of coherent transformations, which ensures the self - regulation of a totality.

Through the structure concept, as an abstract model, the rules governing the transformations and ensuring a system functionality become rationally intelligible.

In the methodological strategy of structuralism, the rule of diachronic, variation enables explaining the system variations by structural invariants.

A distinction is made between "synchronic" designating the relationship between successive terms, therefore structural analysis lies in a topological and relational approach.

Applying the structural analysis rules, and first of all the immanence rule, the analysis is exclusively focused inside the domain under investigation, operating temporarily, for methodological reasons, a closing of the respective domain.

The internal structure of a domain of knowledge is established not on the basis of resemblances but of differences, by grouping and ordering differences more exactly binary opposition, where the relations between the elements are of complementarity.

1.3. The Principle of Sufficient Reason

Leibniz elaborates the principle of sufficient reason and formulates it as follows: "The meaning of sufficient reason (Raison suffisante) is that no fact can be considered true or sufficient and no sentence can be considered true without the existence of a sufficient motivation for why it is like this and not otherwise".

Schoppenhauer consecrated to this principle the paper entitled "The Quadruple

Root of Sufficient Reason", in which he distinguishes the following forms of this principle: the principle of sufficient reason of existence, of becoming, of knowledge and of action, involving the following aspects: existence, cause, knowledge and motive.

1.4 Scientia Generalis

Leibniz, by elaborating "characteristica universalis", i.e. "a general system of signs and formulae" so that in a certain scientific system to each object relationship corresponds a sign, believed in the possibility of constructing a general science.

Within the frame of this science, named "scientia generalis", the principles of the "general methodology" of sciences can be elaborated.

1.5 Semantic Steps

The theory of semantic steps in Semiotic starts from the fact that there are objects, properties and relations which belong to objective reality approached to as a knowledge field.

The objects of the first step which have a corresponding formalization in an object language, constitute the so - called zero steps.

The languages from the second step on, will be called metalanguages and they serve to the formalization of objects on superior steps. The objects, properties and relations of the zero step form the basis of the whole sequence of steps of human knowledge

However, from the theory of types, it follows that any property belongs to a higher step than the objects having that property.

1.6. The Intention And Extension Of Notions

Any notion has two fundamental determinations which are connected, namely the extension and the intension of the notion.

Notions reflect classes of things. The reflection of a class of objects in a notion is called notion extension.

The intension of a notion means the abstract reflection of the invariable properties and relations of a certain class of object.

1.7 Immediate Inferences. Syllogisms

Immediate inferences or inferences lie in obtaining a new sentence from a single proposition. *According to Aristotle a syllogism consist in inferring a sentence from other two sentences.*

1.8 Arborescent Graph. Taxonomy

An arborescent graph is a particular graph in which there is a peak called root, so that any peak of the graph of the graph is linked with S by a unique route.

The arborescent graph is also known as tree. Taxonomy or taxonomic arborescent graph is a graph in which there are inherited proprieties.

The construction of a taxonomy enables the system to know that an element has, besides its own properties, the properties of all its precursors in the graph. Taxonomy is used for a hierarchized graph.

1.9. The Relationship Between Structure And Genesis

Structural analysis constitutes a starting point to a historical analysis and from a genetic perspective, structure itself becomes comprehensive; the dialectic method is seen as unity of structural - functional analysis of historical - genetic analysis implying the study of the origin and evolution of the corresponding structures as the historical product of a self-governing equilibrating process, structural coherence emerging not as static reality but as dynamic virtuality.

Structural analysis correlated with historical - genetic analysis explains the transition from one structure to another.

Each system has a definite structure that includes the resources of surpassing itself.

1.10. The Knowledge Basis

The knowledge basis can be considered as a n - dimensional topological space, on which a geometry can be defined and within it concepts of open sets and contact neighbourhood, frontier, continuity and topological transformation are operand.

The metric space of the knowledge basis should not be limited only to the level of forms detected in the real space but to that of notions.

The information stored in the knowledge basis must be organized in sets or classes

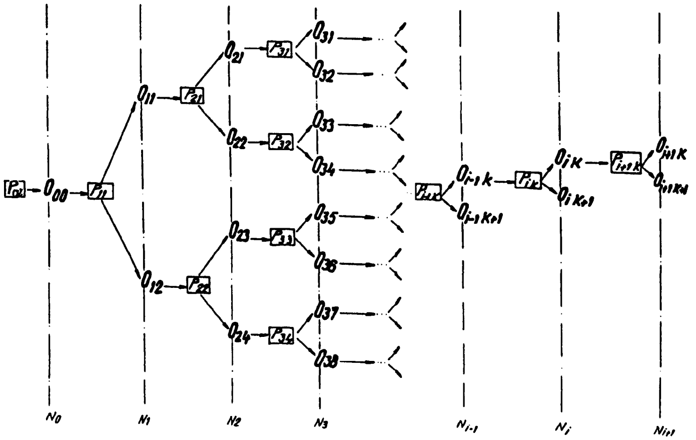


Fig.1. Generalized arborescent hyper graph (taxonomy)

(O_{ik}); the totality of classes O_{ik} forms the knowledge or the references structure of intelligent systems, there being the relation:

$$O_{ik} \neq O_{ik+1}$$
$$O_{ik} \cap O_{ik+1} = \phi \qquad (2)$$

for any O_{ik}, O_{ik+1}, where ⌀ is a void set.

In real conditions, relation (2) is not observed and the class delimitation is vague, the sets defining the classes are "fuzzy" in Zadeh's opinion or "fluid" according to the definition given Gentilhome.

The classes $\{O_{ik}\}$ in the knowledge basis are not equivalent, there being a level sequence of classes of increasing power beginning with the uniques and ending with the reference set $\{O_{00}\}$.

An algebra of relations can be defined of the knowledge basis for the forms of intellectual activity; the relations can realize an application of the knowledge basis in the knowledge basis enable its description under the form of a graph.

2. Modelling Syllogistic - Inferential Processes by Diachronic Syllogistic Structures

2.1. Methodological Principles

1° The principle of convergence to "Axiomatic One". Any structure tends and converges ascending to "Axiomatic One" which is associated to the object O_{00} (fig. 1).
2° The principle of "Sufficient Divergence". Any structure as descending divergent on an unlimited number of levels; for it to be intelligible, a finite number of descendence levels is sufficient.

2.2. Generalized Syllogistic Hyper Graph

The generalized syllogistic graph represents the model of the most general diachronic structures, constituted of structure - diachronic cells; it is elaborated by superposing three arborescent structures:

1) of properties
2) of diachronic and synchronic relations
3) of objects

2.3. The Diachronic Space

Descartes was the first to raise the problem of the coordinate besides of space and time. This paper introduce an extra coordinate besides space and time, namely diachrony, and we shall name diachronic hyper Cartesian space the limitless ensemble of diachronic levels, consisting of a sequence of levels (N_i); each diachronic level has a corresponding "step" in the becoming of the UNIVERSE.

The references system of diachronic space contains the axes of diachrony and syncrony.

The axis of diachrony represents the history of the becoming UNIVERSE, and the axis of synchrony represents UNIVERSE S existing in space time.

2.3.1 The universal parameters of diachronic space

The diachronic space is characterized by the following universal parameters:
1° - diachronic levels (N_i);
2° - the quantity of information corresponding to level (I_i);
3° - level probability (p_i);
4° - equivalence class or level cardinal number (n_i)

$$n_i=2^{N_i}$$
$$p_i=\frac{1}{2^{N_i}} \qquad \textbf{(3)}$$
$$I_i=\log_2 2^{N_i}$$

According to the principle of "SUFFICIENT DIVERGENCE" the last level in the "SUFFICIENT" number of levels, will be associated to the zero step from the theory of semantic steps (N_i).

Considering the objects on the diachronic levels to be sets, applications: Π_i:U⟶U/ρ_i defined by $\Pi_i=\rho_i$, where U is an universe (a set of sets), ρ_i are unique equivalences specific to named cannonical projection of equivalences ρ_i.

Each diachronic level has a corresponding class of equivalence, therefore a cardinal number (n_i).

According to the principle of convergence to "AXIOMATIC ONE", the limit of the cardinal numbers row tends to "ONE".

As cardinal numbers are classes of equivalence, which imply binary equivalence relations defined in Cartesian spaces, the diachronic space constitutes a "hyper cartesian" space.

2.4 Structural diachronic cell

The structural diachronic cell will be defined as a minimal quadri - property, three - objects, three - relation set (fig. 2).

Mathematically, the structural - diachronic cell is defined as the set of the three minimal property - object - relation sets, as follows:

$$\{\{P_{i-1k},P_{ik},P_{i+1k},P_{i+1k+1}\},\{o_{i-1k},o_{ik},o_{ik+1}\},$$
$$\{\mathfrak{R}_{ik\ i-1k},\mathfrak{R}_{ik+1\ i-1k},\mathfrak{R}_{ik\ ik+1}\}\} \qquad \textbf{(4)}$$

The structural diachronic cell can be modelled mathematically by three elementary matrices:

1.- the property - elementary matrix

$$\begin{bmatrix} P_{i-1\ k} & \\ P_{ik} & \\ P_{i+1\ k+1} & P_{i+1\ k+1} \end{bmatrix} \qquad \textbf{(5)}$$

2. - the object - elementary matrix

$$\begin{bmatrix} O_{i-1} & \\ O_{ik} & O_{i\ k+1} \end{bmatrix} \qquad \textbf{(6)}$$

3. - the relation - elementary matrix

$$\begin{bmatrix} \Re_{ik\ i-1k} & \Re_{ik+1\ i-1k} \\ [R_{ik\ ik+1}] & \end{bmatrix} \qquad \textbf{(7)}$$

2.5 The triangle of the three logic principles

From figure 2 the structural - diachronic cell it can be noticed that being given a precursor object O_{i-1k} (a UNIVERSE or a set of sets), it can be divided into two and only two successor objects, O_{ik} and O_{ik+1} (two sets of sets) according to the principle of the "excluded tertiary"so that the two descendant objects (successor) should necessary be in a relation of contradiction according to the principle of the "EXCLUDED

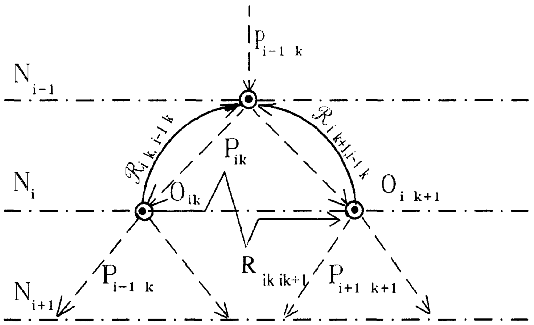


Fig. 2 Structural - Diachronic Cell

CONTRADICTION"; leaving aside property P_{ik}, between the two (resulting) objects there is a relation of equivalence according to the "IDENTITY" principle (fig.3).

2.6 Diachronic interpretation of the definition

In mathematical logic, the notion of definition was introduced by the symbol "$=_{Df}$" placed between two symbolic expressions, specifying that the two symbolic expressions are represented by the term "definiens" and the term "definiendum".

The notion of definition was accepted in mathematical logic in a vague and unprecise manner.

B.Russel had to affirm "the definition is not definable and it is not even a definite notion".

We shall consider that the sign "$=_{Df}$" as a sign of definition is a relation between the expression that defines (definiens) and the defined expression (definiendum), relation which can be true or false.

Assimilating "genus proximum" with the object O_{i-1k} and "differentia specifica" with the property P_{ik} (in the syllogistic hyper graph) and denoting definiendum by D and definiens by d, the definition relation:

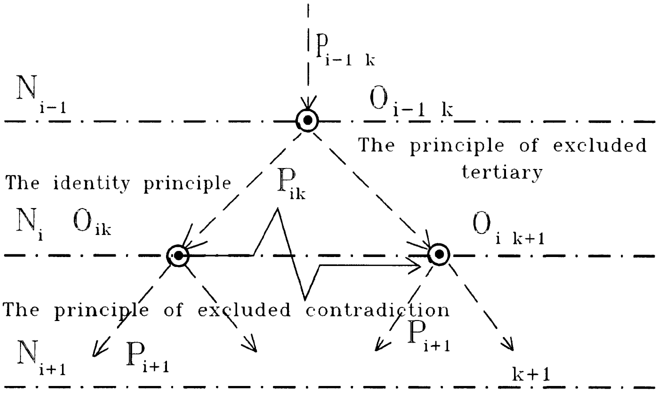


Fig. 3 The triangle of the logic principles

$D=_{Df}$ d, can be interpreted in the following way

$O_{ik}=_{Df}O_{i-1k}$, if all the elements of the objects O_{ik} have the property P_{ik} (fig.4).

It can be noticed from fig.4 that the definition implies 2 diachronic levels (N_{i-1}, N_i), two objects (O_{i-1k} and O_{ik}) and a property P_{ik} and it has an ascendent direction.

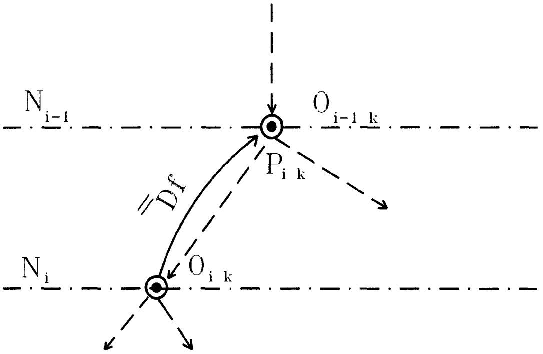


Fig. 4 Diachronic interpretation of the definition

2.7 Quadri Dimensional Interpretation of Syllogisms/Inferences

Let us consider the following inferences/syllogisms.

1. If all spruce firs are Plants an if all Plants are Organisms then all the spruce firs are Organisms.

2. If all the Philosophers are People and if all the People are Social Beings then all Philosophers are Social Beings, by replacing the notions comprised in the two inferences by symbols, the general inferences shall be obtained:

If all S are M
and if all M are P
then all S are P.

Associating the notion ALL by a cardinal number or an equivalent class, it results that notions S, M, P, can be associated with a sequence of three objects (O_{i-1j}, O_{ik}, O_{i+1k}) in a diachronic structure (fig. 5).

From fig.5 it results that the general inference can be interpreted quadri dimensionally as it implies:

1 - three diachronic levels: N_{i-1}, N_i, N_{i+1};

2 - three objects: O_{i-1k}, O_{ik}, O_{i+1k};

3 - three properties: P_{i-1k}, P_{ik}, P_{i+1k};

4 - three relations: SaM, Map, SaP (a the vowel in "afirma" wich replaces the notion "are").

The general inference can be stated as follows: if all the elements of the object O_{i+1k} (set) on the diachronic level N_{i+1} have the property P_{i+1k}, and if all the elements of the object O_{ik} (set) on the level N_i have the property P_{ik} then all the elements of the object $O_{i+1\,k}$ have the property $P_{i-1\,k}$.

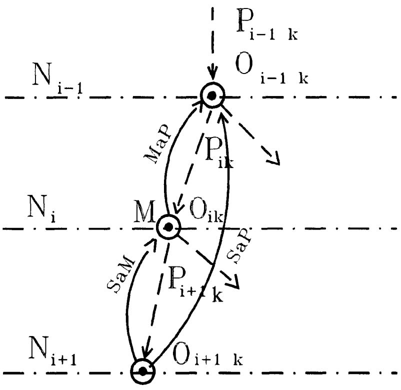


Fig. 5 QUADRI-DIMENSIONAL interpretation of the syllogisms/inferences

Therefore, the syllogism can be represented by a taxonomic arborescent structure in which the elements posses beside their own proprieties all the proprieties of the precursor in the graph. Returning to the two concrete inferences and writing down:

Pruce trees - M
Plants - P and respectively
Organisms - O

Philosopher - F
People - Man
Sociable beings - FS;

we shall obtain the following diachronic structures respectively QUADRI DIMENSIONAL interpretations (fig.6).

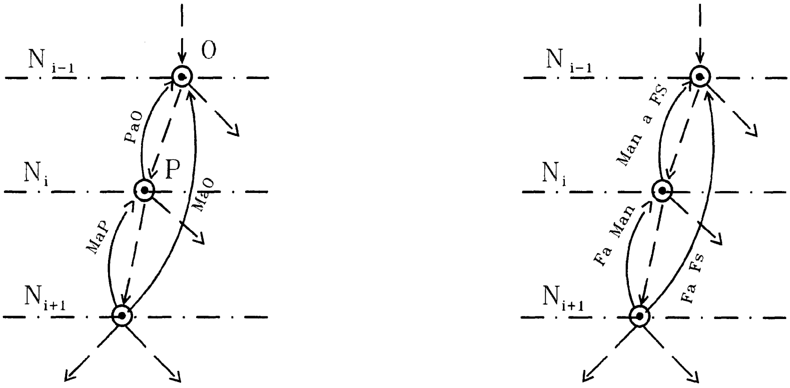


Fig. 6 Diachronic structures

By assimilating notions to object classes, syllogistic figures, respectively Aristotelian syllogistic modes can be modelled, using the graph theory. To each syllogistic mode it corresponds a mathematic model wich is represented by an oriented graph of binary three type arborescent structure (graphs of syllogistic figures No 1).

From the analysis of the representation by graphs of syllogistic modes corresponding to Aristotelian syllogistic figures it can be noticed that a syllogism in a diachronic structure occupies three and respectively four diachronic levels (N_{i-1}, N_i, N_{i+1}, N_{i+2}), among the objects on the various levels the following relations being established:

1. direct diachronic relations - binary relations between two objects on two successive diachronic levels ($\mathfrak{R}_{i-1\ k,ik}$);

2. transcendental diachronic relations - binary relations between two objects which are not on two successive diachronic levels (e.g. $\mathfrak{R}_{i+1\ k,\ I-1k}$);

3. synchronic relations of contradiction - binary relations between two successive synchronic objects to the same diachronic level ($R_{ik,\ ik+1}$);

4. The QUADRI DIMENSIONAL interpretation of each syllogistic mode implies by necessity the existence of a number of diachronic levels, properties, objects and relations (Table 1 corresponding to the four Aristotelian syllogistic figure 1).

E.g. the Barbara syllogistic mode implies three diachronic levels and the perpetual - objectual -relational interpretation:

$$\begin{array}{ll} \mathfrak{R}_1=O_{i,k} @ O_{i-1,k} & \mathfrak{R}_3=O_{i+1,k} @ O_{i-1,k} \\ \mathfrak{R}_2=O_{i+1,k} @ O_{i,k} & P_{i-1,k}\bigcap P_{i,k}\Rightarrow P_{i+1,k} \end{array} \qquad \textbf{(8)}$$

- the BARBARI syllogistic mode implies 4 diachronic levels and propertual - objectual - relational interpretation:

$$\begin{array}{ll} \mathfrak{R}_1=O_{ik}@O_{i-1,k} & \mathfrak{R}_s^{/}=O_{i+2,k}@O_{i-1,k} \\ \mathfrak{R}_2=O_{i+1,k}@O_{i,k} & P_{i-1,k}\bigcap P_{i,k}\bigcap P_{i+1.} \\ \multicolumn{2}{c}{\mathfrak{R}_3=O_{i+1,k}@ O_{i-1,k}} \end{array} \qquad \textbf{(9)}$$

Conclusions

1. In elaborating a Universal Knowledge Basis it is necessary to associate notions and concepts with the objects (Q_{ik}) of the generalized syllogistic hyper graph.

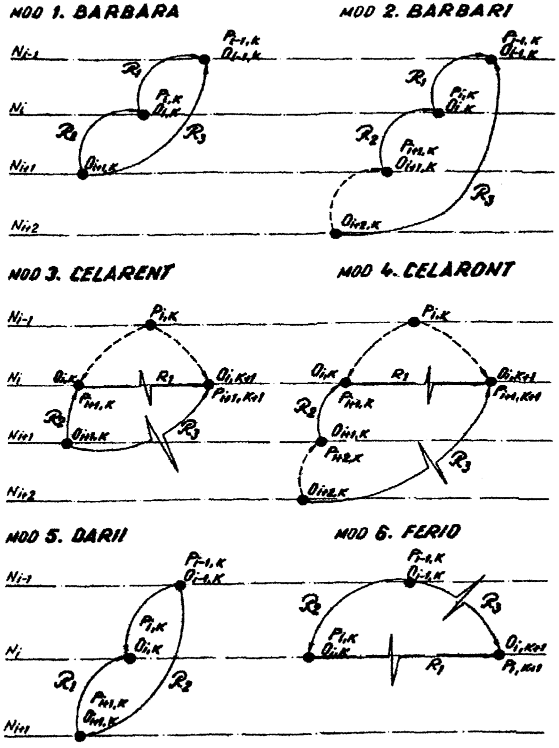
2. The actual (P_{ik}), aprioric ($P_{i-1,k}$) and aposteriori ($P_{i+,k}$) properties correspond to the objects on the diachronic levels; the properly ordered sets of properties, objects and relations constituted in syllogistic rows correspond to a route in the generalized syllogistic hyper graph.

3. The structure of the generalized semantic network of objects demands a hierarhical organization of objects (concepts, notions) by a generalized, taxonomic arborescent structure.

SYLLOGISTIC FIGURE No. 1

TABLE 1

CRT. NO.	SYLLOGISTIC MODES	RELATIONAL - OBJECTUAL PROPERTATIONAL INTERPRETATION
1.	MOD 1 BARBARA	$\mathcal{R}_1 = O_{i,K} \textcircled{a} O_{i-1,K}$ $\mathcal{R}_2 = O_{i+1,K} \textcircled{a} O_{i,K}$ $\mathcal{R}_3 = O_{i+1,K} \textcircled{a} O_{i-1,K}$ $P_{i-1,K} \wedge P_{i,K} \longrightarrow P_{i+1,K}$
2.	MOD 2 BARBARI	$\mathcal{R}_1 = O_{i,K} \textcircled{a} O_{i-1,K}$ $\mathcal{R}_2 = O_{i+1,K} \textcircled{a} O_{i,K}$ $\mathcal{R}_3 = O_{i+1,K} \textcircled{i} O_{i-1,K}$ $\mathcal{R}_3' = O_{i+2,K} \textcircled{a} O_{i-1,K}$ $P_{i-1,K} \wedge P_{i,K} \wedge P_{i+1,K} \longrightarrow P_{i+2,K}$
3.	MOD 3 CELARENT	$R_1 = O_{i,K} \textcircled{e} O_{i,K+1}$ $\mathcal{R}_2 = O_{i+1,K} \textcircled{a} O_{i,K}$ $\mathcal{R}_3 = O_{i+1,K} \textcircled{e} O_{i,K+1}$ $P_{i+1,K} \wedge \overline{P_{i,K}} \longrightarrow \overline{P_{i+1,K+1}}$
4.	MOD 4 CELARONT	$R_1 = O_{i,K} \textcircled{e} O_{i,K+1}$ $\mathcal{R}_2 = O_{i+1,K} \textcircled{a} O_{i,K}$ $\mathcal{R}_3 = O_{i+2,K} \textcircled{a} O_{i+1,K}$ $\mathcal{R}_3' = O_{i+2,K} \textcircled{e} O_{i,K+1}$ $P_{i+2,K} \wedge P_{i+1,K} \wedge \overline{P_{i,K}} \longrightarrow \overline{P_{i+1,K+1}}$
5.	MOD 5 DARII	$\mathcal{R}_1 = O_{i+1,K} \textcircled{a} O_{i,K}$ $\mathcal{R}_2 = O_{i-1,K} \textcircled{i} O_{i+1,K}$ $\mathcal{R}_3 = O_{i-1,K} \textcircled{i} O_{i,K}$ $P_{i+1,K} \wedge P_{i-1,K} \longrightarrow P_{i,K}$
6.	MOD 6 FERIO	$\dot{R}_1 = O_{i,K} \textcircled{e} O_{i,K+1}$ $\mathcal{R}_2 = O_{i-1} \textcircled{i} O_{i,K}$ $\mathcal{R}_3 = O_{i-1,K} \textcircled{o} O_{i,K+1}$ $P_{i-1,K} \wedge P_{i,K} \longrightarrow \overline{P_{i,K}}$



GRAPHS OF THE SYLLOGISTIC FIGURE No. 1.

Concepts, notions and scientific assertions are currently presently in an entropic and redundant manner, making it difficult for the human brain to learn and understand them and at the same time impossible to implement them in Intelligent Systems, respectively in Artificial Intelligence.

References

1. Aristotle. Organon II. Prime Analytics. Scientific Publishing House, Bucharest.
2. Leibniz, G.W. Monadologie, Fragmente zur Logik, Berlin.
3. Klauss, G.(1977). Modern logic.Scientific and Enciclopedic Publishing House, Bucharest.
4. Wang, H. (1972). Studies of Mathematical Logic. Scientific Publishing House, Bucharest.
5. Mirita. I. (1995). Metaformalization of Aristotelian Syllogistic Inferential Processes with a View to Stimulating Reasonings and Thinking Processes Ratiocinations, 14th International Congress of Cybernetics, Namur (Belgium), August 21 - 25.

Complexity of Precedence Graphs for Assembly and Task Planning

João Rocha[1]; Carlos Ramos[1]; Zita Vale[2]

[1]Institute of Engineering - Polytechnic Institute of Porto (ISEP/IPP)
Rua São Tomé, 4200 PORTO, PORTUGAL
Email: {csr, jsr}@dei.isep.ipp.pt
[2]Faculty of Engineering, University of Porto (FEUP)
Rua dos Bragas, 4099 PORTO CODEX, PORTUGAL

Abstract. This paper deals with a complete and correct method to compute how many plans exist for an assembly or processing task. Planning is a NP-hard problem and then, in some situations, the application of time consuming search methods must be avoided. However, up to now the computation of the exact number of alternative plans for any situation was not reported elsewhere. Notice that the complexity of the problem does not depend on the number of involved operations, components or parts. The complexity of the problem depends on the topology of the precedences between operations. With the method presented in this paper, it will be easy to decide the search method to use, since we know how many possible plans could exist before applying the search method.

1. Introduction

In our previous work on the generation of plans for assembly and manufacturing tasks[1] [2 - 4] we used precedence graphs as a structure to represent the involved process. Precedence relations are obtained from constraints. For manufacturing tasks, the main constraints are the process, feasibility and geometric constraints. The constraints imply precedence relationships to guarantee that operations will be executed in the correct order.

In precedence graphs, the manufacturing or assembly operations are the nodes while the precedence relations between operations represented by directed arcs. In this way, we can represent very complex assemblies or manufacturing processes without any combinatory explosion problem in the graph representation since in these graphs the number of nodes is equal to the number of operations. This is important because in some other representation structures (e.g. AND/OR graphs [5]) there is a huge amount of nodes when the number of involved operations increases. However, notice that independently of the used representation the number of the possible plans to execute the task is the same.

[11] Readers interested in a classification of Assembly and Task Planning may consult reference [1].

Although precedence graphs are unable to represent all possible sequences (e.g. mutually exclusive chains), if one sequence is represented, then all feasible plans can be generated. Due to the complexity of the problem (planning is a NP-hard problem [6]) a great amount of time may be necessary to generate all plans for a specific task. For this reason, we decided to implement several algorithms to generate plans using precedence graphs.

One possibility is to generate all plans, but this can be a time consuming work when a great amount of plans exists. Notice that the number of plans does not directly depend on the number of operations, precedence relations, and parts. With one million of operations represented by a sequence of nodes there is only one plan, while with only 10 operations without any precedence (in this case the graph has 10 nodes and no arcs) we have 10! (3 628 800) possible plans. Thus, the number of plans depends on the topology of the Precedence Graph.

Another possibility is the use of heuristics for the fast generation of one or more plans. Here the quality of the solution is not guaranteed and it is strongly dependent of the example being used. However, the generation time is low.

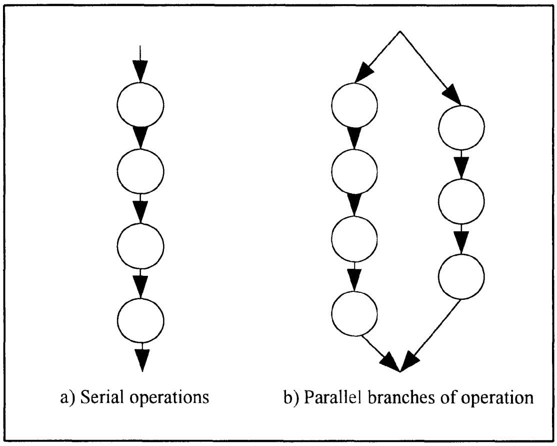
Nevertheless, a very interesting possibility is the generation of the best plan according to time execution criterion. This algorithm takes some more time than the previous one but the best solution according that criterion is achieved. It is important to say that this algorithm is also able to generate the N best plans (N specified by the user). Notice that the best plan according to time execution criterion could not be the best according to other criteria (e.g. line balancing, quality, etc) but with more than one plan the user may impose other constraints on the achieved solutions.

As we described before we have three alternative methods: to generate all plans, to generate one or more plans quickly, and to generate the best plans according to a specific criterion. However, how could we automatically decide which plan generation method will be used? As we show before the complexity is not dependent on the number of operations or on the number of precedences. For this reason, we decided to study and to implement a method to analyze the precedence graph and to compute the number of possible plans without the need to generate all the plans. This method gives, almost immediately, the correct number of possible plans for the task.

A very interesting work in this area is presented by Wolter [7]. Several key aspects of assembly planning systems are analyzed and evaluated. Although several enumerative data structures designed to represent large set of assembly plans are described, precedence graphs are not included. Another difference with the work that is presented in this paper is that only exact upper bounds on size of each structure describe is calculated and not the exact number of plans that could be generated. Finally, only assembly plans are considered in Wolter's analysis.

2. The Slot-Block Theory

The calculus of the number of plans is done in a way that reminds how to calculate the equivalent resistance of an electric circuit. It is necessary to identify serial operations or parallel operations and in each case, we can calculate how many plans with N operations exist. Grouping step by step we will achieve the equivalent, in our case the number of possible plans for the task.


a) Serial operations

b) Parallel branches of operation

Fig. 1. - Main topological elements of precedence graph

Notice that here a plan is understood as a sequence of operations. If parallelism in the plan execution is allowed then some plans may correspond to the same situation.

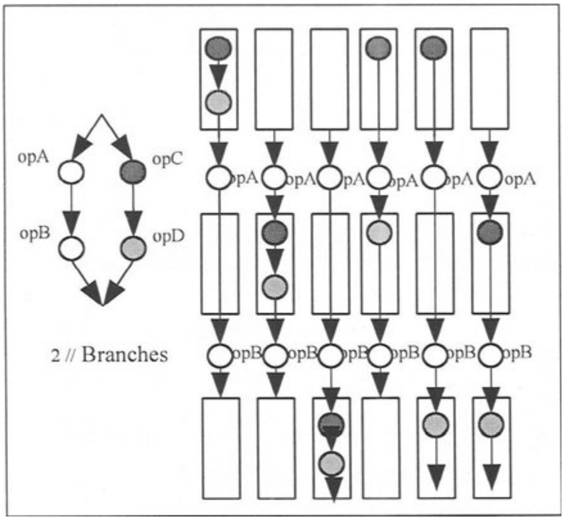


Fig. 2. - Possible plans for precedence graph with 2 parallel branches

As a very simple example, consider three operations (op1, op2 and op3), 2 precedence relations (op1 before op3, op2 before op3) and 3 machines (one for each operation). In this case, the plan {op1, op2, op3} and the plan {op2, op1, op3} are in fact the same. However, if we have only one machine for the three operations these plans are different and may lead to different execution times. This is very simple to understand. For example, consider that the machine has 2 tools (Tx for op2 and Ty for op1 and op3) and that Tx is installed on the machine, for this situation the second plan is more efficient due to the setup time consideration.

Fig. 1 illustrates the main topologic elements found in precedence graphs: serial operations and parallel branches of operations. As the result of each grouping, we will have the number of possible plans (Nplan) with a number of involved operations (Nop).

Fig. 2 illustrates the possible plans for two parallel branches with two operations each. The parallel grouping will give as result 6 plans (Nplan = 6) with 4 operations and helps to understand the basic idea to obtain the number of plans. In any case, we will have 4 operations for the plan since all operations are necessary. Our goal is to compute how many plans will exist without generating them.

The basic idea is to expand one branch on another one. For example considering the left branch of the figure we will have three places (slots) where the operations of the right branch can be placed (before both operations, between op1 and op2 and after both operations). These slots are represented by rectangles in the possible plans. Generally, if we have N operations in one branch we will have N+1 slots (see **Fig. 3**).

Thus, it is necessary to distribute the **m** operations of the other branch in the **n+1** slots of the branch holding the slots. However, one must remember that the precedences in the branch with **m** slots must be guaranteed. The question here is how many different blocks of operations may be placed into the slots.

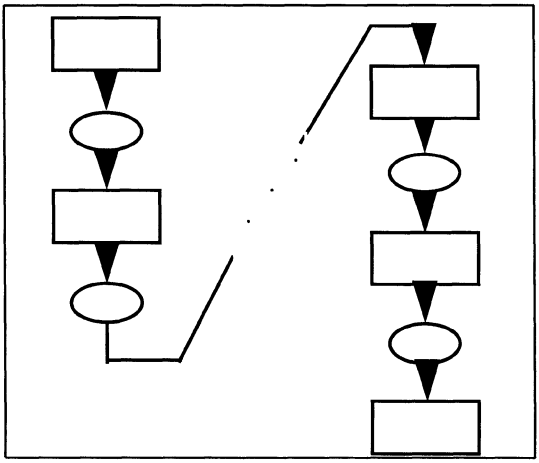


Fig. 3. - Slots in a branch

Table 1 shows the possibilities of distribution of 6 operations grouped by 4 blocks. In the table we consider that op_i precedes op_{i+1}.

Theorem: *For grouping m operations in i blocks, we will have* C_{i-1}^{m-1} *groups* (1)

Proof: from statistical analysis theory, it is known that if we want to group *n* different elements in *k* groups, we have n!/(n-k)! groups. In our case, however, since the precedences in the branch must be guaranteed, the order of the elements in the groups is not important (in fact, only one possible order is allowed). So, the number of possible groups is then n!/k!(n-k)! = C_k^n (again from statistical analysis theory). In our case, with *m* operations we will have *m - 1* transitions from one operation to another one and, with *i* blocks, we will have *i – 1* transitions from one block to another. The problem resumes, then, to calculate the number of combinations between the possible number of places for the transitions (*m – 1*) and the number of transitions that we must have (*i – 1*) → C_{i-1}^{m-1}.

Table 1. - Grouping 6 operations in 4 blocks

1	$\{op_1\}$ $\{op_2\}$ $\{op_3\}$ $\{op_4,op_5,op_6\}$
2	$\{op_1\}$ $\{op_2\}$ $\{op_3,op_4,op_5\}$ $\{op_6\}$
3	$\{op_1\}$ $\{op_2,op_3,op_4\}$ $\{op_5\}$ $\{op_6\}$
4	$\{op_1,op_2,op_3\}$ $\{op_4\}$ $\{op_5\}$ $\{op_6\}$
5	$\{op_1\}$ $\{op_2\}$ $\{op_3,op_4\}$ $\{op_5,op_6\}$
6	$\{op_1\}$ $\{op_2,op_3\}$ $\{op_4\}$ $\{op_5,op_6\}$
7	$\{op_1\}$ $\{op_2,op_3\}$ $\{op_4,op_5\}$ $\{op_6\}$
8	$\{op_1,op_2\}$ $\{op_3,op_4\}$ $\{op_5\}$ $\{op_6\}$
9	$\{op_1,op_2\}$ $\{op_3\}$ $\{op_4,op_5\}$ $\{op_6\}$
10	$\{op_1,op_2\}$ $\{op_3\}$ $\{op_4\}$ $\{op_5,op_6\}$

Theorem: *For placing the blocks in the slots, we will have* C_i^{n+1} *possible combinations.* (2)

Proof: the i blocks can be placed into the slots. Since we have n + 1 slots, it is easy to conclude that for i blocks we have C_i^{n+1} possible combinations.

Thus, for expanding one branch with m operations, grouped in i blocks, on another branch with n operations we have $C_i^{n+1}C_{i-1}^{m-1}$ possible combinations. However, since the number of blocks i vary from 1 to the minimum between *m* and *n + 1*, the total amount of combinations of operations in two parallel branches is:

$$\sum_{i=1}^{\min(n+1,m)} C_i^{n+1}C_{i-1}^{m-1} \text{ combinations (subplans) with n+m operations.}$$

The used reasoning was to expand the branch with m operations on the branch with n operations. The inverse reasoning could be used (to expand the branch with n operations on the branch with m operations). This corresponds to the right-hand side of the following expression that can be mathematically proved:

$$\sum_{i=1}^{\min(n+1,m)} C_i^{n+1}C_{i-1}^{m-1} = \sum_{i=1}^{\min(m+1,n)} C_i^{m+1}C_{i-1}^{n-1}$$

For the example of **Fig. 2** we have m = 2, and n = 2. Thus, the number of possible plans with these 4 operations is given by:

$$\sum_{i=1}^{\min(n+1,m)} C_i^{n+1}C_{i-1}^{m-1} = \sum_{i=1}^{2} C_i^3 C_{i-1}^1 = C_1^3 C_0^1 + C_2^3 C_1^1 = 3*1+3*1=6$$

The first product of the sum corresponds to the possibilities of grouping in one slot (3 possibilities: before opA and opB; between opA and opB; and after opA and opB) the two operations of the right branch in one block (only one {opC,opD}). This part corresponds to the following plans (the 3 first plans of **Fig. 2**): 1 - {opC, opD}, opA, opB / 2 - opA, {opC, opD}, opB / 3 - opA, opB, {opC, opD}

The second product of the sum corresponds to the possibilities of grouping in two slots (3 possibilities: using the first and second slots; using the first and third slots; and using the second and third slots) the two operations of the right branch in two blocks (only one {opC}, {opD}). This part corresponds to the following plans (the three last plans of **Fig. 2**): 1 - {opC}, opA, {opD}, opB / 2 - {opC}, opA, opB, {opD} / 3 - opA, {opC}, opB, {opD}.

3. The Parse Tree

In order to compute the total number of possible plans of a precedence graph we must group operations in serial and parallel until the equivalent has been obtained. A parse tree is built to drive the grouping process.

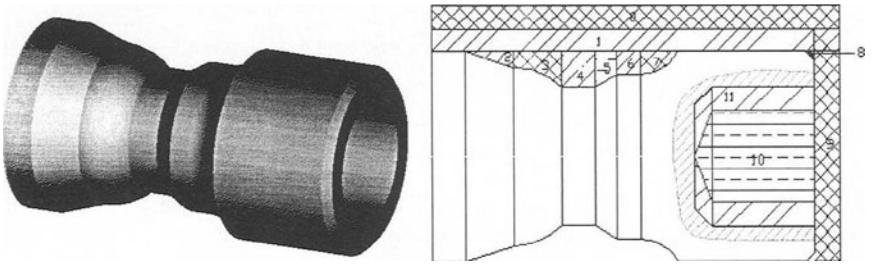


Fig. 4. – an example

Let us consider an example for making the object shown in **Fig. 4**, involving 12 operations, starting with an aluminum cylinder. A lathe can be used to perform the necessary processing operations. The surface of the cylinder must be uniformly adjusted and some facing and turning operations are necessary. By default, we do not know the order by which the operations must be done. The sequence of operations will be obtained by the feasibility and geometric constraints between parts and tools. There are also some processing constraints.

The precedence graph for the above example is shown in **Fig. 5**, considering all feasibility, geometric (e.g. operation 1 must be done before operation 2) and processing constraints (e.g. for this type of material we must adjust the surface before starting turning).

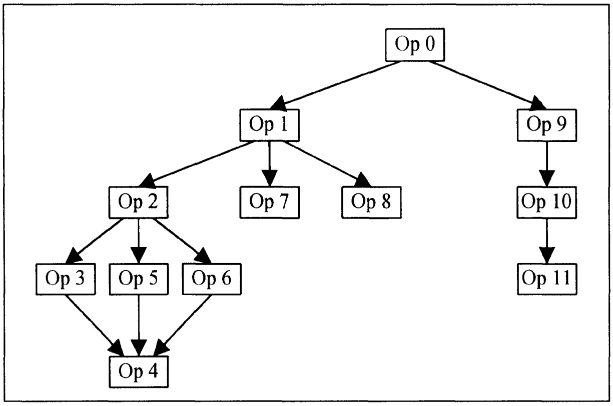


Fig. 5. - Precedence graph for the example

The parse tree for this graph is illustrated in , where the terminal nodes are the operations, **p** represents parallel operations and **s** represents serial operations. The parse tree defines how to proceed in the grouping process to obtain the total number of possible plans. For this example, the number of possible plans is obtained as follows:

- for parallel of {op5, op6}, n=1, m=1 (let us name it as α)

$$\sum_{i=1}^{\min(n+1,m)} C_i^{n+1} C_{i-1}^{m-1} = \sum_{i=1}^{1} C_i^2 C_{i-1}^0 = C_1^2 C_0^0 = 2*1 = 2$$

this means 2 plans (α = 2) with 2 operations (m + n = 2)

- for parallel of {op3, α}, n=1, m=2 (let us name it as β)

$$\alpha * \sum_{i=1}^{2} C_i^2 C_{i-1}^1 = = \alpha * C_1^2 C_0^1 + C_2^2 C_1^1 = 2*(2*1+1*1) = 2*3 = 6$$

this means 6 plans (β = 6) with 3 operations (n + m = 3).

Notice that it is necessary to multiply the sum by α, since the sum represents the possible solution for 1 plan of the α branch; however for α we have 2 plans.

- for the serial operations *op2*, *op4*, and those of , we have the same number of plans than in β, but with 5 operations, since the β parallel has 3 operations
- for parallel {op8, s(op2, op4, β)}, n=1, m=5 (let us name it as χ)

$$\beta * \sum_{i=1}^{2} C_i^2 C_{i-1}^4 = = \beta * C_1^2 C_0^4 + C_2^2 C_1^4 = 6*(2*1+1*4) = 6*6 = 36$$

this means 36 plans (χ = 36) with 6 operations (m+n = 6)

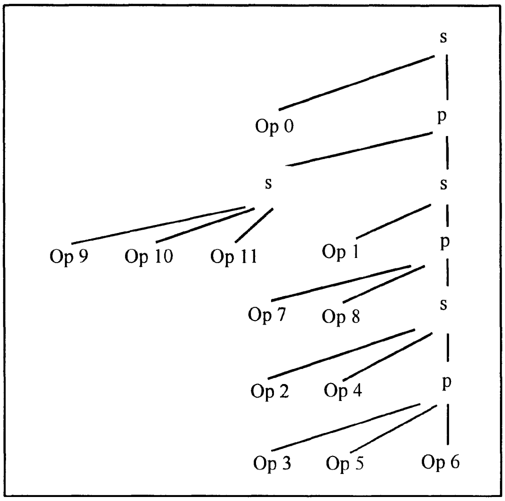


Fig. 6. - parse tree

- for parallel of {op7, χ}, n = 1, m = 6 (let us name it as δ)

$$\chi * \sum_{i=1}^{2} C_i^2 C_{i-1}^5 = \chi * C_1^2 C_0^5 + C_2^2 C_1^5 = 36*(2*1+1*5) = 36*7 = 252$$

this means 252 plans with 7 operations (n + m = 7)

- for the serial operations *op1* and those of δ, we have the same number of plans of δ, but with 8 operations, since the δ parallel has 7 operations
- for the serial operations *op9*, *op10* and *op11* we have only one plan with 3 operations
- for parallel of {s(op9, op10, op11), s(op1, δ)}, n=3, m=8

$$1*\delta*\sum_{i=1}^{4} C_i^4 C_{i-1}^7 = = \delta * C_1^4 C_0^7 + C_2^4 C_1^7 + C_3^4 C_2^7 + C_4^4 C_3^7 =$$

$$= 252*(4*1+6*7+4*21+1*35) = 252*165 = 41580$$

Therefore, since this is the last grouping, the total number of possible plans for the task is 41 580.

Once the number of plans has been achieved, one may decide which policy to use in order to generate the plans. If the number of plans is reduced then the generation of all plans is possible in order to give to the user the possibility to choose by him in a set of few solutions. If the number of plans is high then one may decide to adopt one of the following policies: the fast generation of N plans or the generation of the best N plans according with the time execution criterion. In this way, the user will have the possibility to analyze only a small number of good possibilities. The experience shows that usually the best plan to achieve a compromise of constraints is in the set of the best plans according to the time execution criterion.

4. Execution Plan

The plan generation is achieved considering all precedence relations (obtained from constraints) represented by the graph and the policy chosen as described in chapter 3. Note that, in this case, only feasible plans are generated.

The algorithm used to generate the plan (see [2] for more details) consider, for each operation, the time needed for each operation as well as the time for changing from one operation to another (e.g., change the tool, wait a certain time before a specific operation due to the involved process and so on). In this algorithm, the configuration of the manufacturing system is also considered, since the sequence of operations is important due to the time needed for changing from one operation to another.

Then it is possible to select the best plan (if that policy was chosen) considering time execution as criterion, since all times involved in the operations, tools' changing and setup are considered. In the example, as the number of different operations is small and the number of total operations is bigger, the main problems (concerning choice of best algorithm) are not the operations time, but the tools' changing and setup times. This is why the best plan is the one that minimizes the number of tools' changing operations and times needed to move tools from one operation to another.

The next step is the generation of the program. The configuration of the manufacturing system must also be considered, since programs should be generated for each machine. This corresponds to the areas of Computer Aided Process Planning and CAD/CAM. The reader can find some good references of this kind of works in [8-12].

5. Conclusions

In this paper, we have presented a method for computing the complexity of a precedence graph for manufacturing or assembly tasks. As the core of the method we created the slot-block theory, that allows to group branches of precedence graphs in order to know how many possible combinations of operations (subplans) exists. A parse tree defines how to proceed in the grouping process to obtain the total number of possible plans of a precedence graph.

The method presented in this paper is implemented in software. Using the software one can rapidly obtain the number of plans of a specific precedence graph, and decide which plan generation algorithm to use. The decision about which algorithm to use depends on the total number of plans as well as on the purposes of the planning system (on-line versus off-line, strategic versus tactical or reactive).

This work shows the impact of Information Technology on industrial processes. Besides the computer aided process planning part of our work that is not described here (see references [2 - 4]) we introduced a new method for the complexity analysis of precedence graphs.

Further work on this method will be to consider several resources to execute the manufacturing and assembly task. In this case, some different plans are equivalent due to the parallelism during task execution. Temporal reasoning is important for this kind of study. Another future direction is the integration with work being developed in the area of production planning and scheduling [13-14].

6. Acknowledgements

The authors of this paper would like to acknowledge FLAD (the Portuguese-American Foundation for the Development) for the support of Project 443/93 and JNICT (the Joint National Institute for Scientific Research and Technology) for the support of Projects 2556/95 and 2551/95.

References

1. S. Gottschlich, C. Ramos, D. Lyons: Assembly and Task Planning: A Taxonomy. IEEE Robotics and Automation Magazine, 1(3): 4-12, September 1994
2. J. Rocha, C. Ramos: Plan Representation and Heuristic Generation of Operation Sequences for Manufacturing Tasks. In International Conference on Data and Knowledge Systems for Manufacturing and Engineering - DKSME'96, pages 223-242, 1996
3. J. Rocha, C. Ramos: Plan Representation and Generation for Manufacturing Tasks. In First IEEE International Symposium on Assembly and Task Planning, pages 22-27, 1995
6. J. Rocha, C. Ramos: Task Planning for Flexible and Agile Manufacturing Systems. In IEEE/RSJ International Conference on Intelligent Robots and Systems - IROS'94, 1994
7. L. Homem de Mello: Task sequence planning for robotic assembly. PhD Dissertation, Carnegie Mellon University, 1989
8. S. Gupta, D. Nau: Optimal Block's World Solutions are N-P. Technical Report, Computer Science Department, University of Maryland, 1990.
9. J. Wolter: A Combinatorial Analysis of Enumerative Data Structures for Assembly Planning. Journal of Design and Manufacturing, 2(2): 93-104, 1992
10. C. C. Hayes: MACHINIST: A Process Planner for Manufacturability Analysis, 1995
11. J. Tulkoff: CAPP. Computer Aided Process Planning. Computer and Automated Systems Association of SME, 1985
12. L. Figueiredo: Conversion and Execution of Process Plans (in portuguese). Master Thesis, University of Porto, 1996
13. A. Kusiak: Intelligent Manufacturing Systems. Prentice-Hall International Series in Industrial and Systems Engineering, 1990
14. D. Das, S. Gupta, and D. Nau: Reducing cost set-up by automated generation of redesign suggestions. In ASME Computers in Engineering Conference, 1994
15. C. Ramos, A. Almeida, Z. Vale: Scheduling Manufacturing Orders with Due Dates acting as real deadlines. Advanced Summer Institute of Network of Excellence in Intelligent Control and Integrated Manufacturing System, ASI'95, 1995
16. C. Ramos: An Architecture and a Negotiation Protocol for the Dynamic Scheduling of Manufacturing Systems. In IEEE Int. Conference on Robotics and Automation - ICRA'94, pages 3161-3166, 1994

A Monte Carlo Algorithm for the Satisfiability Problem

Habiba Drias

USTHB, Institut d'Informatique,
BP 32 El-Alia Bab-Ezzouar, 16111 Alger Algerie
Fax: +213 2 515779, E-mail: drias@ist.cerist.dz

ABSTRACT Recently a randomized algorithm based on Davis and Putnam Procedure was designed in [16] for the purpose of solving the satisfiabilty problem. In this letter another Monte Carlo algorithm following from an original algorithm [4] is proposed. The average performance of the algorithm is polynomial and the probability that the algorithm fails to yield a correct answer for some data is less than ε. Results are compared with those given in [16] and show an interesting performance for our algorithm.

1 Motivation and preliminaries

The satisfiability problem or SAT for short is of special interest because it has many applications notably in theorem proving, automated reasoning etc... Since SAT is NP-complete, a great amount of research works tempt to lessen the difficulty of the problem. Cook in the second Turing Award lecture on Computational Complexity has recommended the use of probability to palliate the combinatory explosion of NP-complete problems [1]. Probabilistic algorithms try to bring an answer to the intriguing open question whether P=NP. The class of problems solved by this type of algorithms in polynomial time is called RP(for Random Polynomial). We know that $P \subseteq RP \subseteq NP$ assuming $P \neq NP$. If we can prove $P = RP$ and show that any given NP-complete problem is in RP then we conclude $P = NP$. Unfortunately both of these premises are still open questions. The results given in [6,11], when combined all together yield that no polynomial average time algorithm exists for the region $c(\frac{\ln n}{n})^{\frac{3}{2}} < p < (t\frac{\ln n}{n})^{\frac{1}{2}}$ where $k \leq n^t$, n and k being respectively the number of variables and the number of clauses of the instance, c and t are constants and p the probability that a literal appears in a clause. The findings in [4] improve these results in the sense that this region is reduced to $c(\frac{\ln n}{n})^{\frac{3}{2}} < p < (1-r^{\frac{-1}{b}})^{\frac{1}{2}}$ for SAT-instances and in case $r < (1 - t\frac{\ln n}{n})^{-b}$, where r is the largest length of clauses and b a constant.

Perhaps, the most known algorithm for solving satisfiability is the Davis and Putnam procedure or DPP for short. Randomized Davis and Putnam procedure exists in the literature [16]. The author in the specified paper takes advantage of previous results on probabilistic analyses of DPP especially those described in [6] and proposes a randomized approach for solving SAT. The expected performance of this algorithm is polynomial to n.

The present study represents a logical continuation of the work presented in [4]. A randomized algorithm with a polynomial average time complexity is developed for SAT. For a subset of instances, the algorithm yields the right answer whereas for the remaining ones, it may be defective and produce a result with a probability of failure not exceeding ε. The results of the algorithm is compared to those of [16] and show a better performance.

1.1 The satisfiability problem

Let $[x_1, x_2, ..., x_n]$ be a set of Boolean variables. The set of literals is then $[x_1, x_2, ..., x_n, \overline{x}_1, \overline{x}_2, ..., \overline{x}_n]$, $\overline{x}_i$ denotes the negation of x_i. A clause is a Boolean disjunction of literals. The length of a clause is defined to be the number of literals involved in the clause. A set of k clauses over n variables represents an instance of satisfiability, namely SAT(k,n). r_i denotes the length of the i^{th} clause. When all clauses have the same length equal to r, the instance is denoted r-SAT(k,n). An example of an instance of satisfiability can be:

$$(\overline{x}_1 + \overline{x}_2 + \overline{x}_5)(x_1 + \overline{x}_2)(\overline{x}_2 + \overline{x}_4)(\overline{x}_1 + x_2 + \overline{x}_3)(\overline{x}_1 + \overline{x}_2 + \overline{x}_3 + x_4)$$

This instance is denoted SAT(5,5). A solution to SAT is an assignment of Boolean values to variables such that every clause is satisfied. $(x_1 = 0, x_2 = 0, x_3 = 1, x_4 = 0, x_5 = 1)$ is a solution for the instance given in this example.

1.2 The Probabilistic model

The probabilistic model considered in this paper is the known constant density model $M(k, n, p)$ due to Goldberg [9] and used in many papers such as [6,7,10,11,12,13,14]. A clause is constructed by including a positive literal with probability p ($p < \frac{1}{2}$), a negative literal associated with the same variable with the same probability.

Let us consider two different clauses c_i and c_j. A variable in either positive or negative form appears in both clauses with probability equal to p^2 and appears in none of the two clauses with probability equal to $(1 - p)^2$. It appears in just one clause, in either c_i or c_j, with probability equal to $2p(1 - p)$. Hence the probability that a variable does not appear simultaneously in both clauses is equal to $2p(1 - p) + (1 - p)^2 = 1 - p^2$.

2 The Monte Carlo algorithm

The randomized algorithm presented in this section derives from an algorithm, namely E-SAT for counting and checking satisfiability designed recently [4]. It is worth describing the latter even in a succinct way.

2.1 Algorithm E-SAT

Algorithm E-SAT responds to the decision satisfiability problem. The clauses of the instance of satisfiability are put in a matrix, namely A. A cell of A may contain a literal of a clause or be empty. A is then converted into another equivalent matrix B whose cells hold terms, which are conjunction of literals. The format of these terms allows the enumeration of solutions of the instance. The example below illustrates the conversion of the instance given in the previous example.

$$\mathrm{A} = \begin{bmatrix} \overline{x}_1 & \overline{x}_2 & \overline{x}_5 & - \\ x_1 & \overline{x}_2 & - & - \\ \overline{x}_2 & \overline{x}_4 & - & - \\ \overline{x}_1 & x_2 & \overline{x}_3 & - \\ \overline{x}_1 & \overline{x}_2 & \overline{x}_3 & x_4 \end{bmatrix}$$

Each row of A represents a clause. The symbol – denotes an absence of literal. Let us modify A as follows:

$$B = \begin{bmatrix} \overline{x}_1 & x_1\overline{x}_2 & x_1x_2\overline{x}_5 & - \\ x_1 & \overline{x}_1\overline{x}_2 & - & - \\ \overline{x}_2 & x_2\overline{x}_4 & - & - \\ \overline{x}_1 & x_1x_2 & x_1\overline{x}_2\overline{x}_3 & - \\ \overline{x}_1 & x_1\overline{x}_2 & x_1x_2\overline{x}_3 & x_1x_2x_3x_4 \end{bmatrix}$$

$B[i,j]$ is the conjunction of $\overline{A}[1,j], \overline{A}[2,j], \ldots, \overline{A}[i-1,j]$ and $A[i,j]$. A row of B can be viewed as an ORing of Boolean terms or in other words as a disjunctive form. The fourth row of B for instance holds the Boolean expression $\overline{x}_1 + x_1x_2 + x_1\overline{x}_2\overline{x}_3$ where $+$ denotes the OR Boolean operator. The equivalence between a row of B and its corresponding row in A can be proved using the absorption law of Boolean algebra $(x + \overline{x}y = x + y)$.

Algorithm E-SAT proceeds by crossing matrix B in a depth-first order. A vertical path corresponds to a term obtained by merging a term from each row of B. The term if non null, that is, if it does not involve a variable in both positive and negative forms, represents a group of solutions.

Algorithm E-SAT

INPUT: an instance of satisfiability

OUTPUT: 'Satisfiable' if the instance is satisfiable, 'unsatisfiable' otherwise

METHOD: It is described by the following steps:

1- Preprocess the instance clauses [3] and put them in
matrix A

2- convert matrix A into matrix B as it is shown in the example

3- cross matrix B in depth-first order

4- when a non null final term is found, output 'satisfiable' and stop

5- if no non null final term exists then output 'unsatisfiable' and stop

The preprocessing of instance clauses is discussed in [4] and consists in rearranging positions of literals and clauses in order to cut off early potential non solutions paths and then finding a non null solution path faster. In fact, literals are placed from left to right according to the decreasing order of their frequency in the instance. Such placement guaranties early abortion of non solutions paths during the search.

The main program of E-SAT consists of a call to procedure *satisfiable.*

```
var satisf : Boolean;
begin
  satisf := false;
  if satisfiable({},1) then output ('satisfiable') else output ('unsatisfiable')
end;

procedure satisfiable( var $S$: set of literals; $i$: integer);
begin
    $j := 1$;
    while ($j \le r_i$) and ( not satisf) do
        begin
        $S = S \cup B[i,j]$;
        if $S$ non null then
            if $i = k$ then satisf := true else satisfiable($S, i+1$);
        $S := S - B[i,j]$;
        $j := j + 1$;
        end
        return(satisf)
end;
```

S, the first parameter of procedure satisfiable holds a partial solution path. When it becomes null after the insertion of a term, we suppress the latter from S and add the next one according to the depth-first order. The process is iterated until finding a solution or reaching the end of the search.

It has been shown that the average time complexity of algorithm E-SAT is $O(kr^m)$ for the class of SAT(k,n) instances verifying $p \geq (1 - r^{\frac{-1}{b}})^{\frac{1}{2}}$, where r is the largest length of clauses, m and b are constants and $b = \frac{m(m-1)}{2}$ [4]. The originality of this algorithm comes from the fact that its performance is polynomial for a class of instances depending on r unlike the previous works where this class is defined by the parameter n and k.

2.2 Algorithm rand-SAT

The randomized algorithm, namely rand-SAT is based on E-SAT. It chooses at random m vertical paths and tests whether they correspond to groups of solutions whereas in E-SAT, the search for solutions paths is dictated by the depth-first order. Since in rand-SAT a limited number of paths are tested for possible solutions, the answer may be mistaken. In compromise the reduction of consulted paths makes the algorithm run faster. In E-SAT all paths are checked until finding out a non null term or exploring all the alternatives. This yields an exact solution with an exponential time complexity for some cases.

Let us now turn to the details of rand-SAT. If one of the vertical paths chosen at random from B corresponds to a non null term thus to groups of solutions, the algorithm outputs 'satisfiable' and stops. On the contrary if none of the drawn paths denote a group of solutions, the algorithm outputs 'unsatisfiable' and the probability of making an error is ε.

Algorithm rand-SAT

INPUT: an instance SAT(k,n) and ε a probability of making an error

OUTPUT: the right answer 'satisfiable' or a likely mistaken answer 'unsatisfiable' with probability of failure equal to ε

METHOD: it is described by the following steps:

```
1- Preprocess the clauses and put them in matrix namely, $A$
2- convert matrix $A$ into matrix of type $B$
3- set $m = \frac{\ln \varepsilon}{\ln(1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}})}$
4-for $i = 1$ to $m$ do
        4.1 choose at random from $B$ a vertical path $S = c_1, c_2, ..., c_j, .., c_k$
        $1 \leq c_j \leq r_j$    $1 \leq j \leq k$, then $sol = B[1, c_1]B[2, c_2]...B[k, c_k]$
        4.2 if $sol$ is non null then (* if sol does not involve opposite literals*)
                4.2.1 answer 'satisfiable'
                4.2.2 stop
5- answer 'unsatisfiable' with probability of failure equal to $\varepsilon$
6- stop
```

2.3 Probability of error and polynomial average time

The probability of error that may be produced by algorithm rand-SAT is specified by the following theorem.

Theorem 1 *Let m be the number of iterations of the loop in rand-SAT. If* $m \geq \frac{\ln \varepsilon}{\ln(1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}})}$ *then the probability that algorithm rand-SAT yields an incorrect result is less than* ε*.*

Proof Let $S = \{c_1, c_2, ..., c_k\}$ be a vertical path $1 \leq c_j \leq r_j$ $1 \leq j \leq k$. The probability that a path S is non null is given in [4] by the formula:

$$(1-p^2)^{\sum\limits_{i=2}^{k}(i-1)c_i + \sum\limits_{i=2}^{k}\sum\limits_{j=1}^{i-1} c_j - \frac{k(k-1)}{2}}$$

The probability that it is null is equal to

$$1-(1-p^2)^{\sum\limits_{i=2}^{k}(i-1)c_i + \sum\limits_{i=2}^{k}\sum\limits_{j=1}^{i-1} c_j - \frac{k(k-1)}{2}}$$

And the probability that m generated paths are null is

$$(1-(1-p^2)^{\sum\limits_{i=2}^{k}(i-1)c_i + \sum\limits_{i=2}^{k}\sum\limits_{j=1}^{i-1} c_j - \frac{k(k-1)}{2}})^m$$

If all the m drawn paths are null whereas the instance is satisfiable, rand-SAT fails to answer correctly with a probability equal to

$$(1-(1-p^2)^{\sum\limits_{i=2}^{k}(i-1)c_i + \sum\limits_{i=2}^{k}\sum\limits_{j=1}^{i-1} c_j - \frac{k(k-1)}{2}})^m \leq (1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}})^m$$

where r is the largest length of clauses.

Let prove now that

$$m \geq \frac{\ln \varepsilon}{\ln(1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}})} \Longrightarrow (1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}})^m \leq \varepsilon$$

$$m \geq \frac{\ln \varepsilon}{\ln(1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}})} \Longrightarrow \ln \varepsilon \geq m \ln(1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}}) \Longrightarrow$$

$$\varepsilon \geq (1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}})^m \qquad \square$$

Let show now that the average complexity of rand-SAT is polynomial.

Theorem 2 *rand-SAT takes polynomial time on average.*

Proof We will prove that the average number of iterations of the loop of rand-SAT is polynomial when $p < (1 - r^{-\frac{1}{b}})$. When $p \geq (1 - r^{-\frac{1}{b}})$, rand-SAT is polynomial since E-SAT is polynomial.

Let $m = \frac{\ln \varepsilon}{\ln(1-(1-p^2)^{\frac{(2r+1)k(k-1)}{2}})}$

If $s = \frac{(2r+1)k(k-1)}{2}$ then

$$\mathrm{m} = \frac{\ln \varepsilon}{\ln(1-(1-p^2)^s)}$$

$$= \frac{\ln \varepsilon}{\ln(1-(1-\binom{s}{1}(p^2)+\binom{s}{2}(p^2)^2 \;-...))}$$

$$== \frac{\ln \varepsilon}{\ln(\binom{s}{1}(p^2)-\binom{s}{2}(p^2)^2 \;+...)}$$

$$\binom{s}{1}(p^2) - \binom{s}{2}(p^2)^2 \;+ ... \leq \binom{s}{1}(p^2) + \binom{s}{2}(p^2)^2 \;+ ... = \sum u_i$$

$$\frac{u_{i+1}}{u_i} = \frac{(s-i)p^2}{i+1} \leq 1 \text{ for } i_0 \geq \frac{s-p^2}{p^2+1} \geq \frac{s-1+r^{-\frac{1}{b}}}{2-r^{-\frac{1}{b}}} = t \text{ when } p < (1-r^{-\frac{1}{b}})$$

$$\sum u_i \leq s\binom{s}{t}(p^2)^t \leq s^2(p^2)^t$$

$$m \leq \frac{\ln \varepsilon}{\ln s^2(p^2)^t} = \frac{\ln \varepsilon}{2\ln s+t\ln(p^2)}$$

$$\leq \frac{\ln \varepsilon}{2\ln s+t\ln(1-r^{-\frac{1}{b}})} \approx \frac{\ln \varepsilon}{2\ln r+4\ln k+t\ln(1-r^{-\frac{1}{b}})} \leq \frac{\ln \varepsilon}{t\ln(1-r^{-\frac{1}{b}})} \qquad \square$$

It is easy to see that the expected performance of rand-SAT, which is polynomial is interesting knowing that the average complexity of the randomized algorithm presented in [16] is O(k$^{\beta}$), β being a constant.

3 Conclusion

A Monte Carlo algorithm, namely rand-SAT, for solving the satisfiability problem is presented in this paper. Its average performance is compared with previous works in the literature and denotes an interesting behavior of the algorithm.

4 References

[1] S.A. Cook. An overview of computational complexity. *Communication of the ACM.* 26, 6(1983)

[2] M. Davis and H. Putnam. A computing procedure for quantification theory. *J Assoc Comput Mach* 7 (1960) 202-215

[3] H. Drias. Number of Solutions for SAT instances: A Probability Distribution Study. *J info Science and Engineering* 8, 4 (1992) 621-631

[4] H. Drias. A new algorithm for counting and checking satisfiability problem and a restricted unfavorable region for polynomial average time. *in proc of the 30th IEEE Symposium on System Theory (*1998*)*

[5] O. Dubois and J. Carlier. Probabilistic approach to the satisfiability problem. *Theor Comp Sci* 81 (1991) 65-76

[6] J. Franco. On the probabilistic performance of algorithms for the satisfiability problem. *Info Proc Let* 23 (1986) 103-106.

[7] J. Franco and Y.C. Ho. Probabilistic performance of heuristic for the satisfiability problem. *Discrete Appl Math* 22 (1988/1989) 35-51

[8] M.R. Garey and D.S. Johnson . *Computers and Intractability.* Freeman & C. 1979.

[9] A. Goldberg. Average case complexity of the satisfiability problem. *in proc. Fourth workshop on Automated deduction* (1979) 1-6

[10] A. Goldberg , P.W. Purdom and C.A. Brown. Average time analyses of simplified Davis-Putnam procedures. *Info Proc let* 15 (1982)72-75

[11] K. Iwama. CNF satisfiability test by counting and polynomial average time. *SIAM J Comp* 18, 2 (1989) 385-391

[12] P. Purdom. Search rearrangement backtracking and polynomial average time. *Artificial Intelligence* 21 (1983) 117-133

[13] P. Purdom and C.A. Brown. Polynomial average time satisfiability problem. *Info Sci* 41 (1987) 23-42

[14] P. Purdom and C.A. Brown. The pure literal rule and polynomial average time. *SIAM J Comp* 14 (1985) 943-953

[15] R. Solovay and V.A. Strassen. A fast Monte Carlo test for primality. *SIAM J. Comput* 6 (1977) 84-8

[16] L.C. Wu and C.Y. Tang. Solving the satisfiability problem by using randomized approach. *Info Proc Let* 41 (1992)187-190

Applying the Propose&Revise Strategy to the Hardware-Software Partitioning Problem*

M. L. López-Vallejo[1], C.A. Iglesias[2] **, and J. C. López[1]

[1] Dpto. de Ingeniería Electrónica,
E.T.S.I. Telecomunicación
Universidad Politécnica de Madrid, E–28040 Madrid, Spain
{marisa,lopez}@die.upm.es
[2] Dpto. de Teoría de la Señal, Comunicaciones e Ing. Telemática,
E.T.S.I. Telecomunicación
Universidad de Valladolid, E–47011 Valladolid, Spain
cif@tel.uva.es

Abstract. *Hardware-software Co-design* is a new discipline that provides methodologies and tools for the fast development of complex heterogeneous systems by combining hardware and software development cycles. This paper shows how the *Propose&Revise* PSM can be used to solve the hardware-software partitioning problem, the Co-design task that makes the decision on the best implementation of the different components of a system. In particular, a fuzzy expert system, SHAPES, has been developed following the CommonKADS methodology.

1 Introduction

Hardware-software partitioning is the Co-design task that performs the assignment of components of a system description to heterogeneous implementation units: ASICs (hardware), standard or embedded micro-processors (software), memories, etc. It is the key task in *Hardware-Software Co-design*, because the decisions that are made at this time directly impact on the performance and cost of the final solution. The aim of the partitioning is to find an implementation of the system which fulfills all the specification requirements (functionality, goals and constraints) at a minimum cost. The usual way of addressing this problem is by means of general optimization procedures (mainly heuristic algorithms), for instance [3, 6, 4, 7]. In this paper a knowledge based approach for the partition of complex systems into hardware and software components is proposed.

SHAPES (Software-HArdware Partitioning Expert System) is a fuzzy-logic based expert system which provides an easy way to address hardware-software partitioning. The development and structuring of the system has been done following the CommonKADS [8] methodology, in particular the *Propose&Revise*

* This work has been funded by projects HADA (CICYT TIC97-09 28) and BELSIGN (HCM CHRX-CT94-0459)

** This research was partly done while Dr. Iglesias was visiting the Dep. Ingeniería de Sistemas Telemáticos (Universidad Politécnica de Madrid).

PSM has been used. The tool considers two important aspects of general optimization problems: the possibility of dealing with imprecise and usually uncertain values (by means of the definition of fuzzy magnitudes), and the use of an expert knowledge (the one of the system designer) in the decision making process. This kind of techniques has been previously applied to different and important design automation phases, like analog circuit synthesis [9] or test space exploration [5].

The outline of this paper is as follows. First, the partitioning problem within hardware-software co-design is presented. Section 3 justifies the use of artificial intelligent techniques for hardware-software partitioning. Next section presents an overview of the partitioning model, followed by the description of the system inference structure and its basic modules. Finally, experimental results are presented and some conclusions are drawn.

2 Partitioning as a Key Task in Hardware-Software Co-design

The main objective of design automation tools developers is to provide methods to design high performance systems at the lowest price, two antagonistic goals that should be considered in a practical approach. To reach such an objective, it is necessary to look for the best configuration of possible implementations of the system components. *Hardware-Software Co-design* addresses this problem by combining the hardware and software development cycles and looks for the best tradeoffs among the different hardware and software solutions. It can be said, in a general way, that a hardware implementation requires high costs and a long design cycle, but provides the best performance. On the other side, a software implementation is flexible, not expensive and with shorter design time, but supplies lower performance. The main advantages of using a co-design methodology in the design of heterogeneous systems are the following:

1. It can be found a good tradeoff among flexibility, performance and cost.
2. The design cycle is reduced, what results in a shorter manufacturing process (smaller time to market).
3. The global latency of the system is decreased, since hardware and software parts will execute concurrently. Moreover, the use of a hardware co-processor allows a timing improvement regarding the standard processors performance figures.

Every co-design methodology is complex enough to be quite different from any other. Nevertheless, there are some basic tasks common to all of them. The main tasks involved in the co-design life cycle can be seen in figure 1, and are:

1. High level specification of the system, with languages like VHDL, C, etc. It should include functional and non-functional requirements. Therefore, standard languages must be extended.

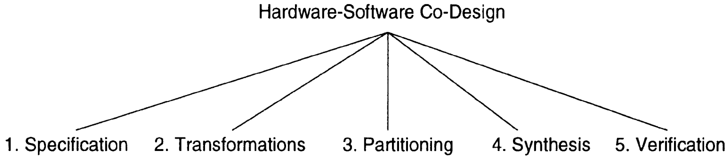


Fig. 1. Task Model of Hardware-Software Co-design and Partitioning

2. Formal transformations and optimizations: for instance, the translation of the initial description into an intermediate format, which eases the handling of information in later steps.
3. Hardware-software partitioning.
4. Synthesis, with three main tasks: the hardware synthesis, the code generation for the software blocks and the interface generation.
5. Verification of the system functionality and constraints, very often done by co-simulation.

3 Rationale for a Knowledge Based Hardware-Software Partitioning

In the traditional design methodology, heterogeneous system development was fully characterized by an initial handmade partitioning phase. The system designer decided which blocks of the system could be implemented in hardware and which ones could be realized in software, taking into account different pieces of estimation and basically his/her knowledge as an expert in the field. To automate this partitioning phase, it is necessary to mimic the way a skilled designer performs this step. In order to achieve such a goal, most approaches use classical optimization algorithms to solve the problem, introducing complicated cost functions with the purpose of combining different and opposite objectives, such as maximizing performance, minimizing area, etc. For example, we can mention the following approaches:

Gupta [6] presents a solution based on the classical heuristic search min-cut. In [4] a simulated annealing algorithm with fine granularity is applied. Eles [3] shows how tabu search provides better results for coarse grain partitioning than simulated annealing. An interesting platform for system partitioning has been developed by Vahid [10], applying classical hardware partitioning algorithms to hardware-software architectures. Finally, a coarse grain algorithm based on list scheduling has been developed in [7].

In this paper, a knowledge based approach that uses the designers' expertise acquired in manual partitioning is proposed. The main benefits of a knowledge-based approach are:

- The designer's knowledge is acquired for automatic handling, while the results obtained by traditional procedures must be reviewed by a designer.
- The whole process can be traced.

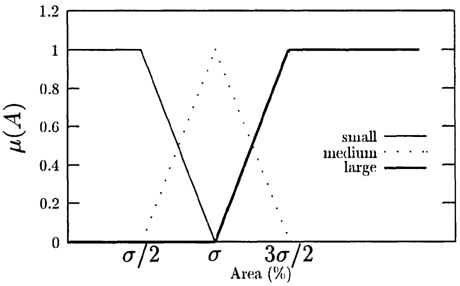


Fig. 2. Linguistic Variable *Area*

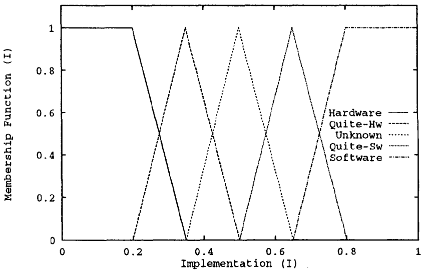


Fig. 3. Linguistic Variable *Implementation*

- The knowledge of the system can be increased as long as the system is used with different cases.

There is also a good foundation to use fuzzy logic:

- Being the designer knowledge imprecise in nature, fuzzy logic can be used to include the subjectivity implicit in the designer reasoning.
- Most data are estimates, which can be easily modeled with fuzzy linguistic variables. For instance, the area estimate of a block is modeled as a linguistic variable with the terms {`small`, `medium` or `large`} (figure 2). Since this value depends on the particular system under evaluation, the linguistic variable has been dynamically defined according to a granularity measure, σ:

$$\sigma = \frac{100}{i}$$

being i the number of tasks to consider.
- Some of the parameters defined along the process can be characterized by their intrinsic looseness. As an example figure 3 shows the implementation value of a task, which reflects the tendency of the task to be implemented as hardware or software. In this case, we can talk for instance about a *quite hardware* task or a *very software* task.

4 Problem Formulation

There are two main issues that must be defined to perform hardware-software partitioning:

1. The target architecture, which will influence the whole process. We have considered a target architecture that follows a classical scheme composed of a processor running the software, an ASIC implementing the hardware and a single bus memory-CPU-peripherals.
2. The description input: from the initial specification we extract a directed and acyclic graph which represents the system execution flow. Following a coarse grain approach, the graph nodes represent tasks and the edges data and control dependencies.

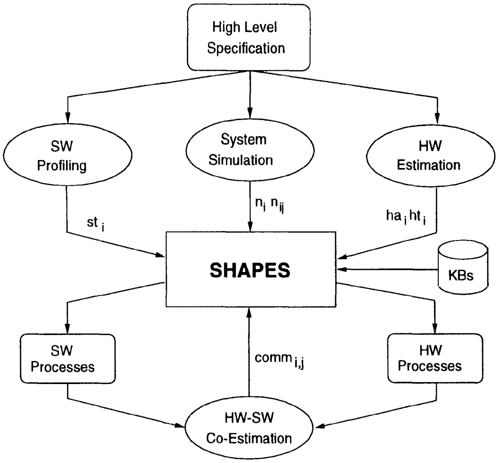


Fig. 4. Sources of information related to SHAPES

Additional information is attached to every graph node. Figure 4 shows how this information is obtained. In detail, for a node i the following data are required: hardware area (ha_i), hardware execution time (ht_i), both calculated by estimation of the synthesis process [2], software execution time (st_i) evaluated by profiling of the all-software solution, and the average number of executions of the task (n_i) obtained after repeated system simulations. Edges have also associated a communication estimate $(comm_{ij})$ obtained from three different values: the transfer time $(t_{trans}(i,j))$, the synchronization time $(t_{synch}(i,j))$ and the average number of times the communication takes place $(n_{i,j})$.

Once partitioning is performed, the output produced by the tool is composed of three main results:

- The assignment of blocks to hardware or software implementation units.
- The schedule of the task graph: starting and finishing time for every task.
- The communication overhead produced in the interface.

5 Partitioning Expertise Model

The CommonKADS methodology [8] has been followed to analyze and design the expert system. The task model of the hardware-software co-design problem is shown in figure 1. After developing an initial organization model, the feasibility of a knowledge based system for the partitioning task has been considered.

The expertise model of the partitioning task has been done using the CommonKADS library [1]. According to this library, the partitioning task has been classified as a configuration task with antagonistic constraints, and the *Propose and Revise*(P&R) problem solving method has been followed, whose inference structure is shown in figure 5. The inference structure is a compound of predefined inference types (how the domain concepts can be used to make inferences,

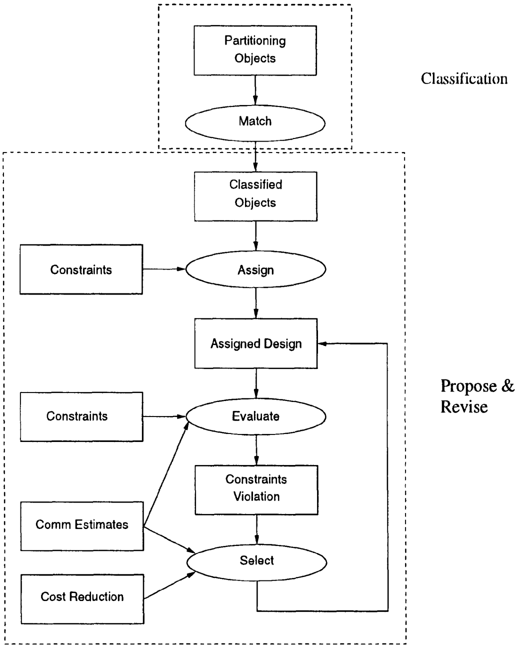


Fig. 5. Inference structure for Hardware-Software Partitioning

represented as ellipses) and domain roles (rectangles) presenting a general view of the PSM and the knowledge used by the inferences. This generic inference structure will be adapted to our problem. An initial *heuristic classification* is carried out before the standard P&R for providing additional information to the propose task.

Following the inference structure, four domain roles have been distinguished: *match* carries out the initial heuristic classification (section 6.1), *assign* that proposes an initial configuration (section 6.2), *evaluate* revises the validity of the solution (section 6.3), and *select* proposes a new configuration (section 6.4). The design of the system has followed a structure-preserving approach, defining a module for each domain role.

6 Partitioning Design Model

6.1 Match (Heuristic Classification)

The first step in SHAPES is to provide additional information for the P&R method, what is performed by a *heuristic classification module.* In this module the observables are the estimates attached to every graph node properly converted into fuzzy variables. In particular, the following variables are considered: (1) hardware area, (2) time improvement obtained when moving the node from hardware to software, and (3) average number of executions per node.

The heuristic rules stored in the knowledge base produce the matching between the input variables and the output solution: the variable `implementation`, shown in figure 3. As it has been previously stated, this variable gives an idea of the intrinsic tendency of a task to be implemented in special purpose hardware or as software running on a standard processor. As an example of the rules of this module, it can be stated:

```
if hw_area is small and time_improvement is high and
   number_executions are not few
then implementation is very hardware
```

It can happen that no rule is activated for a given block, being the implementation unknown. In that case, the tool can be configured to ask the designer an implementation value for the specific block and increase the knowledge base.

A crisp value of implementation is produced to obtain an initial partition. It is necessary to produce output results that will be used in standard tools like the scheduler. This crisp classification output is the set of input blocks ordered by their implementation degree: value 0 stands for hardware and 1 for software.

6.2 Assign

The assign inference provides the first proposal of solution by allocating part of the tasks to hardware and the rest to software. In this module the observables are the system blocks with their related implementation value and the system constraints (maximum hardware area, A, and maximum execution time, T), and produces as output a *threshold* which determines the hardware-software boundary.

The threshold value is obtained after estimating the *"hardness"* of the specification requirements: how critical (or not) the constraints are regarding the extreme values of the system performance (all hardware and all software solutions). Upon this parameter a system configuration is composed. Two linguistic variables are defined from the system requirements: `time_constraint` and `area_constraint`. They both must be dynamically defined because their values are relative to the specific system under study. For instance, `time_constraint` is defined as a function of the *Time Constraint Ratio* ρ_T,

$$\rho_T = \frac{T}{MaxT}, \qquad \text{with} \qquad \frac{MinT}{MaxT} \leq \rho_T \leq 1$$

where MinT is the execution time for the all hardware solution, and MaxT is the execution time for the all software solution.

6.3 Evaluate

The evaluate module computes the different parameters that provide a measure of the quality of the proposal obtained after the assignment phase. This module is not based on knowledge. Here three parameters have been considered:

1. The estimation of the area needed to implement the hardware part A_p.

2. The scheduling of the assigned task graph, which gives the final execution time or latency T_p.
3. The communication costs in the hardware-software boundary.

For the selected hardware modules a rough area estimation is performed by adding the area estimates of all the blocks. Area estimates correspond just to one of the possible points of the design space for a hardware block.

Task scheduling is implemented by means of a list-based scheduling algorithm. The scheduler takes into account the timing estimates of every task in the graph and the dependencies between them. As output it gives the final execution time for that partition, T_p, and the communication cost produced in the hardware-software interface. Again, timing estimates correspond to an average point of the design space. Hardware and software tasks can execute concurrently in the standard processor and the application specific co-processor (ASIC).

The communication costs consider two different aspects: the total number of transfers that occur in the hardware-software interface, and the *communication penalty*, which computes the global delay introduced in the system execution time due to specific communication waiting.

Once these parameters have been calculated, the select inference will check the plausibility of the proposed partition. With this purpose additional information is then originated to know which is the state of the constraints. Specifically, the following variables are defined:

- The area and time overhead (with its linguistic variables `area_gap` and `time_gap`): $\Delta A = A - A_p$, $\Delta T = T - T_p$, being A_p and T_p the actual partition area and latency.
- The communication penalty (detailed in section 6.4).
- The processor and ASIC throughputs (with fuzzy variables `proc_thr` and `asic_thr`), defined as τ_{proc} and τ_{asic}:

$$\tau_{asic} = \frac{\sum_{i \in HW} st_i}{T_p} \qquad \tau_{asic} = \frac{\sum_{i \in SW} ht_i}{T_p}$$

6.4 Select

The select inference implements the most complex inference in SHAPES. It revises the proposed solution and searches for another proposal. Consequently, the select strategy has been divided in two stages:

1. *Diagnosis*: studies how close the solution comes to the optimum and how many corrections are needed. In detail, the following symptoms are computed:
   - The degree of violation or satisfaction of the hardware area and global time constraints (upon their respective gaps).
   - How balanced the proposal is regarding the processor and ASIC Throughputs ($\tau_{proc}/\tau_{asic} \approx 1$).
   - Which tasks are related to the communication penalty.

2. *Operation*: performs the selection of a new proposal based on the previous information and rule base of previous experiences. In fact, two kinds of revisions can be performed. First, if the system constrains are not met the method must refine the solution to find a feasible one. Second, if constraints are met and there is a considerable gap the solution can be refined to reduce costs. There are two non-exclusive alternative strategies to choose:
   - To migrate tasks from hardware to software or vice versa according to the global constraints. This is carried out by tuning the threshold.
   - To improve the partition attending to minimize the communication on the hardware-software boundary.

   As an example of the revise strategies, and due to the important role that the communication cost plays in the design of heterogeneous systems the last point will be detailed in the following.

Communication-based Reordering Attending to the communication penalty computed in the evaluation phase, a new configuration can be proposed. This configuration can be performed by reordering the tasks according to a new parameter: the *Communication Rate*, ρ_{comm}. The *Communication Rate* factor of a software task i is calculated as

$$\rho^{HW}_{comm}(i) = \frac{\sum\limits_{j \in HW} t_{comm}(i,j)}{\sum\limits_{j \in HW} t_{comm}(i,j) + \sum\limits_{k \in SW} t_{comm}(i,k)}, \qquad \text{with} \qquad 0 \leq \rho_{comm}(i) \leq 1$$

This factor considers the communications of the task i with the tasks placed in hardware, being relative to the total number of communications of the task. In the same way, ρ^{SW}_{comm} can be calculated for tasks implemented in hardware. Note that $t_{comm}(i,j) = 0$ if there are no data dependencies between tasks i and j. This factor is estimated and fuzzified for all the tasks that lead to a delay in the hardware-software interface. A new linguistic variable is defined, the `comm_shift`, that have the attributes {`shift_hw`, `balaced` or `shift_sw`}. During the reordering the implementation value of the tasks that contribute to the communication delay is updated. If the communication factor is high and the task is close to the threshold, the kind of implementation can even be changed. For instance, there can be a hardware task whose implementation is `quite-hw` and the communication rate is `shift_sw` (ie. the task communicates more frequently with the software part than with the hardware part). The select module will propose a change of the task implementation (`quite-sw`) providing a new configuration.

7 Case Study

As an example, a task flow graph with 23 nodes will be traced to illustrate the way SHAPES works. Based on the estimates previously attached to the nodes and edges of the graph the *match module* produces the histogram of figure 6. For this graph, the system requirements are bounded by the values:

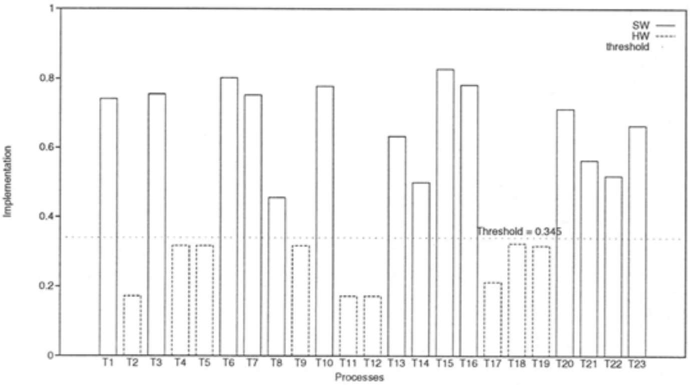


Fig. 6. Classification and assignment results for the case study

- All software execution time, $MaxT = 1276$ t.u. (time units),
- All hardware execution time, $MinT = 460$ t.u. and
- All hardware area, $MaxA = 41{,}570$ e.g. (equivalent gates).

When entering several constraint values (hardware area A and maximum execution time T) the results obtained in the first iteration differ. For example, if the designer fixes the system constraints to $A = 30000$ and $T = 600$, the *assign module* computes a *threshold* of 0.345 and proposes a first partition with 9 hardware tasks and 14 software tasks (figure 6). This partition is characterized with a hardware area $A_p = 22450$ and execution time $T_p = 745$. The revision of the proposal finds that this is not a feasible solution, and analyzes the problem:

- There is a big hardware area gap ($\Delta A = 7550$).
- There is a medium time violation ($\Delta T = -145$).
- The processor and ASIC are unbalaced.
- There is a high communication penalty with 4 tasks involved.

The *select module* performs communication-based reordering by accepting changes from software to hardware. After two revisions, a feasible solution is found ($A_p = 28270$, $T_p = 545$). Table 1 shows some results obtained with other system constraints and the number of iterations needed to find a feasible solution.

8 Conclusions and Future Work

This paper has shown an application of expert systems to hardware-software co-design. One of the advantages of our approach are the declarative specification of the knowledge bases, that contain the expertise of the designers and allow its modification by the user. This tuning of the system rules cannot be carried out following approaches such as simulated annealing or constructive methods. In contrast to the other methods, our system provides traceability of the decisions made during the process.

Constraints		First Iteration			
Area A	Time T	Threshold	Estim. Area	Estim. Time	Iterations
40000	1200	0.15	0	1276	2
40000	500	0.69	28270	504	3
30000	1000	0.28	7640	1017	2
20000	1200	0.21	6110	1088	1
20000	800	0.43	20450	545	3

Table 1. Results obtained for the example with different system requirements

The Propose&Revise PSM has been succesfully applied to perform hardware-software partitioning. Additionally, the use of the CommonKADS methodology has considerably reduced the development time of the system.

The comparison of this system with other approaches is promising. Nevertheless, a rigorous comparison cannot be shown because of the lack of available benchmarks for co-design and the feasibility of comparison among different approaches based on fine or coarse grain.

References

1. J. A. Breuker and W. Van de Velde, editors. *The CommonKADS Library.* Netherlands Energy Research Foundation ECN, Swedish Inst. of Comp. Sci., Siemens, Univ. of Amsterdam and Free Univ. of Brussels, November 1993. Technical report, ESPRIT Project P5248.
2. C. Carreras, J. C. López, M. L. López-Vallejo, C. Delgado-Kloos, N. Martínez, and L. Sánchez. A Co-Design Methodology Based on Formal Specification and High-Level Estimation. In *Proc. Workshop on HW/SW Co-Design*, 1996.
3. P. Eles, Z. Peng, K. Kuchcinski, and A. Doboli. System Level Hardware/Software Partitioning based on Simulated Annealing and Tabu Search. *Design Automation for Embedded Systems*, 2(1):5–32, January 1997.
4. R. Ernst, J. Henkel, and T. Benner. Hardware-Software Cosynthesis for Microcontrollers. *IEEE Design & Test of Computers*, pages 64–75, Dec 1993.
5. M. Fares and B. Kaminsa. Exploring the test space with fuzzy decision making. *IEEE Design & Test of Computers*, pages 17–27, Fall 1994.
6. R. K. Gupta and G. De Micheli. HW-SW Cosynthesis for Digital Systems. *IEEE Design & Test of Computers*, pages 29–41, 1993.
7. A. Kalavade and E. A. Lee. The Extended Partitioning Problem: Hardware/Software Mapping, Scheduling and Implementation-bin Selection. *Journal of Design Automation of Embedded Systems*, 2(2):125–164, March 1997.
8. A. Th. Schreiber, B. J. Wielinga, J. M. Akkermans, and W. Van de Velde. CommonKADS: A comprehensive methodology for KBS development. Deliverable DM1.2a KADS-II/M1/RR/UvA/70/1.1, University of Amsterdam, Netherlands Energy Research Foundation ECN and Free University of Brussels, 1994.
9. A. Trouble, J. Chávez, and L. G. Franquelo. FASY: an fuzzy tool for analog synthesis. *IEEE trans. on CAD*, 15(7):705–714, Jul 1996.
10. F. Vahid and T. Le. Extending the Kernighan/Lin Heuristic for Hardware and Software Functional Partitioning. *Design Automation for Embedded Systems*, 2(2):237–261, March 1997.

Fuzzy Hybrid Techniques in Modeling*

M. Delgado[1], A. F. Gómez-Skarmeta[2], J. Gómez Marín-Blázquez[2], H. Martínez Barberá[2]

[1]Dept. Ciencias de la Computación e Inteligencia Artificial.
Universidad de Granada, Spain
mdelgado@ugr.es
[2]Dept. Informática y Sistemas. Universidad de Murcia, Spain
skarmeta@dif.um.es, ogro@dif.um.es, humberto@fcu.um.es

Abstract. We will describe several methods to approximate the fuzzy rules tuning and generation problem. To generate the rules we will use several clustering algorithms. This approach supposes that the data lacks of structure. To tune the rules we will use two different techniques. One of them is based in descent gradients. The other one is based in a try to tune the rules outputs to reduce the error.

1 Introduction

In the last times, several different techniques, which we may be subsumed in the frame of Soft-computing, have appeared in the literature related with fuzzy modeling. Thus we find proposals that use fuzzy neural networks [1] [2], fuzzy subset theory combined with descent gradient techniques [3], or with clustering techniques [4] [5] [6], etc. Recently it has been appeared combinations of those techniques to try to solve the different problems related with systems modeling [7] [8] [9].

In the context of fuzzy modeling, the use of clustering based techniques had had a great success. The objective of this clustering is to detect the behaviors present in the data obtained in a system under observation with no other additional information, to perform a process of fuzzy modeling through a set of fuzzy rules. Some different approximate and descriptive methods have been proposed in [10] [11]. With the results described in those works, what we get is a rule set that is only a first approach to the problem of fuzzy modeling of the system we study. The next objective is to aboard the problem of how to optimize the rules generated with some of the proposed methods. To do so we propose the use of tuning techniques based in the modification of some of the parameters of the fuzzy rules obtained with the clustering techniques and give sense to the creation of hybrid systems applied to fuzzy system modeling.

In this paper we present several approaches to the problem of fuzzy rules generation using clustering techniques. With those techniques we can create some fuzzy rules without the assumption of some kind of structure in the input data. Such

* This work has been partially supported by CICYT project TIC97-1343-C002-02

rules will be the starting point for a later tuning applying two different algorithms of tuning. This work plan let us find rules with a high approximation degree starting from heterogeneous data.

In the next section we present the notation and the way in which the rules are used to describe the behavior of the system. We do so by developing a method to create fuzzy rules from a clustering. After that, we will describe two different methods to tune such rules to reduce the error. Finally we will show some examples and give some conclusions.

2 Fuzzy Modeling

2.1 Detection of Behavior Clustering

We will consider, without losing generality, a MISO system (Multiple Input, Single Output). We want to find a system that approaches the function $\varphi : X^p \rightarrow Y$ that models the system. We have a sample set of the behavior of the system in the space $(X^p \times Y)$, $\Omega = ((x_{t1}, x_{t2}, \ldots, x_{tp}), y_t)$ $t = 1,2,\ldots,n$, where $X_1, X_2, \ldots, X_p$ are the domains of the inputs and Y is the domain of the output.

We also consider that we know, in some way, the existence of k clusters in the data. It is important to note that the clusters we work with can be directly obtained applying the analysis of fuzzy clustering in the data, or after performing a pre-process using a hierarchical clustering, as have been studied in other works [12]. We don't need to know any other information of the data structure (for example, the linearity or not of them). We will use a clustering algorithm to find k centroids. With the objective of performing a data clustering, and being able to detect in this way the associated centroids to the different detectable clusters, we are going to use the classical Kohonen self-organizative algorithm. So not only is modified in each loop the "winner" centroid or the closest to the learning example but a centroids set of the vicinity, vicinity that will decrease when the number of iterations grows up. An important aspect to stand out is that the clustering process is performed over the input and output product space. The objective of this process is, as is shown in several works [13] [14], to perform a better detection of the clusters that exists in the data. This is so because not only is taken into account the interactivity between input product space but also the consequences of its interrelation with the output space. So the clustering will be performed over the $(X^p \times Y)$, that is, we will also take into account the output values of the examples. In this way, we will obtain k centroids C_h with this form $(w_1, w_2, \ldots, w_p, w_{p+1}), h = 1,2,\ldots,k$ where $w_1, w_2, \ldots, w_p$ belong, respectively, to the domain $X_1, X_2, \ldots, X_p$ and w_{p+1} belongs to the domain of Y.

Supposing that this clustering defines the behavior tendencies of the examples, we want to obtain a characterization of such behavior using centroids to create *k* fuzzy rules that will have this form:

$$R_h : \textbf{If } x \textbf{ is } A_h \textbf{ then } y \textbf{ is } B_h \quad h = 1,\ldots,k \tag{1}$$

where A_h and B_h are, respectively, the fuzzy sets in X^p and Y obtained as a representation of the *h-th* centroid C_h of $\left(X^p \times Y\right)$.Once created all those rules (in the next chapter we will show how we will create them) an inference mechanism can be used for any new input, using an approximate reasoning.

2.2 Fuzzy Rules Generation

For simplicity we have chosen the simplified fuzzy rule form [15] where the fuzzy set of the consequent for each rule is changed for a simple value or singleton consequent, being, in our case, a real number. So we will have rules of this type:

$$R_h : \textbf{If } x \textbf{ is } A_h \textbf{ then } y \textbf{ is } v_h \quad h = 1,\ldots,k \tag{2}$$

the fuzzy sets of A_h are generated with center (or middle point, depending of the kind of fuzzy number we use) equal to the centroid. To calculate the width we will assign to each example a centroid (particularly the centroid closest to the example) and check, for each centroid, among the examples assigned to it, which one is the farthest. The width of the fuzzy set generated by such centroid will always be greater than such distance. The reason to do so is to assure that all examples are members of, at least, one fuzzy set. It have to be greater to avoid that such element (the one that is furthest) have a membership value of zero (see below membership functions), what is equivalent to a no-membership. This will also assure that, at least, one rule will be fired for each example.

The membership value, to which we refer to, is an interpretation of the expression of the antecedent of the rules (*x* is A_h). Particularly, the rules can be expressed as:

$$R_h : \textbf{If } x_1 \textbf{ is } A_{1h} \textbf{ and } x_2 \textbf{ is } A_{2h} \textbf{ and} \ldots \textbf{ and } x_p \textbf{ is } A_{ph} \textbf{ then } y \textbf{ is } v_h \quad h = 1,\ldots,k \tag{3}$$

Each one of the expressions (x_j **is** A_{jh}) are interpreted as *membership degree* $\mu_{A_{jh}}\left(x_j\right)$ of input value x_j to the fuzzy set A_{jh}. Such membership degrees are defined by the membership function that depends of the kind of fuzzy number used.

We will use triangular fuzzy numbers. Two numbers, center (c) and base (b) define such numbers. It membership function is:

$$\mu_{A_{jh}}(x_j)=\begin{cases} 1-\dfrac{2\cdot\left\|x_j-c_{jh}\right\|}{b_{jh}} & c_{jh}-{}^{b_{jh}}\!\!/_2 \le x_j \le c_{jh}+{}^{b_{jh}}\!\!/_2 \\ 0 & \text{otherwise} \end{cases} \tag{4}$$

Each rule needs an initial output value. Such singleton consequent associated to each rule is calculated as the mean of the outputs of the examples that fires it, weighted by the membership value of such example. If we have n examples of p dimension in the form $e=[(x_1,y_1),(x_2,y_2),\ldots,(x_n,y_n)]$ with $x_i \in X^p$ and $y_i \in Y$, the initial rule output value v_h is expressed in this way:

$$v_h=\frac{\sum_{j=1}^{n}\mu_{A_h}(x_j)\cdot y_j}{\sum_{j=1}^{n}\mu_{A_h}(x_j)} \tag{5}$$

where $\mu_{A_h}(x_j)$ is the *firing strength* of the rule R_h for the example x_j. There are several ways to express such firing strength. It can be defined as a measure of the global coincidence of each of the components of the example $x_{ij}, i=1,2,\ldots,p$ with each one of the fuzzy numbers A_{ih} that compose the rule (as we shown before, such coincidence is measured by the membership value). In our case we will use the t-norm product (Eq. 6) of the firing strengths.

$$\mu_{A_h}=\prod_{i=1}^{p}\mu_{A_{ih}}(x_{ij}) \tag{6}$$

Once defined all the rules of the system, we can use them to produce results with new examples. The way to infer such results uses a the Mizumoto's simplified reasoning method [15] and is defined as the mean of the rule outputs of the rules fired by the example weighted for the firing strength of such rule for such example. That is:

$$y=\frac{\sum_{h=1}^{k}\mu_{A_h}v_h}{\sum_{h=1}^{k}\mu_{A_h}} \tag{7}$$

2.3 Fuzzy Rules Tuning

The set of rules generated is already a good approximation to the system to be modeled. We can calculate the error we have when we use them. As we only know the outputs of the examples, we can estimate with them the error we have when we approximate the function. We will use the Mean Square Error:

$$E = \frac{\sum_{j=1}^{n} \left(y_j^* - y_j\right)^2}{n} \tag{8}$$

being y_j^* the real value of the *j-th* example and y_j the fuzzy rules system inferred value for the *j-th* example. Starting with those measures we can try to reduce the values of them by tuning the rules. In particular, we should try that the inferred values get closer to the real ones. To do so we have created two approximations based in the output tunes and in descent gradients.

Method 1

The method 1 is an algorithm that tries to improve the global output by a set of individual improvements. This method modify the v_h of the rules to get a y_j closer to y_j^*.

To fulfill this objective we will get one by one the examples $e_j = (x_j, y_j^*)$. For each example we check how many rules are fired with such example and we will obtain in this way a subset of them $P_j = \left\{R_h / \mu_{A_h}\left(x_j\right) > 0\right\}$. Among all the fired rules, we chose the one with the strongest firing strength $R_k \in P_j / \mu_{A_k}\left(x_j\right) = \mathrm{Max}\left(\mu_{A_i}\left(x_j\right) \in P_j\right)$. We tune the output of such rule to get closer the rule output value v_k to the output value of the example y_j^* weighted by a learning rate $g_S(t)$ with $1 \geq g_S(t) \geq 0$ and weighted also by the fire strength of the rule $\mu_{A_k}(t)$, using the following criteria $v_k(t+1) = v_k(t) + g_s(t) \cdot \mu_{A_k}(t) \cdot \left[y_j^* - y_j\right]$. In this way we achieve that the system global output for such example y_j get closer to y_j^*.

The strong point of this system is its simplicity, but it do not assure that the tune made by an example do not prejudice tunes made by other examples.

Method 2

This second method tries to improve the error using descent gradient. Using the error expression we obtain expressions of the dependence of y_j with regard to the output of the rules (the v_h) and with regard to the own fuzzy numbers that fires them. So we can modify both the v_h and the fuzzy sets of the rules. In particular, a descent method seeks for the vector Z, which minimizes an objective function $E(Z)$ where Z is a p-dimensional vector $Z = (z_1, z_2, \ldots, z_p)$ of the tuning parameters. In this method, the vector that decreases the objective function $E(Z)$ is expressed by $\left(-\partial E/\partial z_1, -\partial E/\partial z_2, \ldots, -\partial E/\partial z_q\right)$, and the learning rule is expressed in the following way:

$$z_i(t+1) = z_i(t) - \mathrm{K}\frac{\partial E(Z)}{\partial z_i}, \qquad (i = 1,2,\ldots,q) \tag{9}$$

where t is the tune process iteration and K is a constant. In our case, the objective function to minimize is the error (Eq. 8) of the fuzzy rules system.

If we use triangular fuzzy numbers, as we described before, we see that the system depends of three different groups of variables: the centers of the fuzzy numbers (c_{ij}), the base of the fuzzy numbers (b_{ij}) and the consequents of the rules (v_i) with $(i = 1,2,\ldots,n)$ and $(j = 1,2,\ldots,p)$. So the vector Z have the form $(z_1, z_2, \ldots, z_q) = (c_{11}, \ldots, c_{np}, b_{11}, \ldots, b_{np}, v_1, \ldots, v_n)$. Developing we obtain:

$$c_{ij}(t+1) = c_{ij}(t) - \mathrm{K}_c \cdot \frac{\partial E}{\partial a_{ij}} \tag{10}$$

$$b_{ij}(t+1) = b_{ij}(t) - \mathrm{K}_b \cdot \frac{\partial E}{\partial b_{ij}} \tag{11}$$

$$v_i(t+1) = v_i(t) - \mathrm{K}_v \cdot \frac{\partial E}{\partial v_i} \tag{12}$$

The rest is to perform an iterative process of application of these three expressions on the described variables. The process will finish when the reduction of the error between two iterations is less than a determined value. The constants that appear in the expressions let us tune the sharpness of the method.

3 Simulation Results

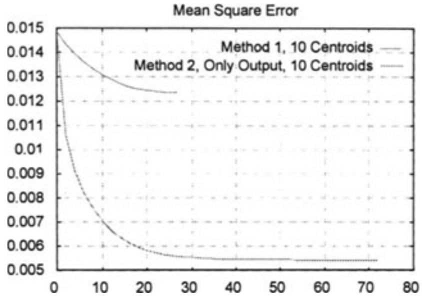
We have verified both methods described before with the non-linear Sinc function:

$$z = Sinc(x,y) = \frac{Sin(x)}{x} \cdot \frac{Sin(y)}{y} \tag{13}$$

As example group we got 225 random samples from an area inside the interval $[-10,10]\times[-10,10]$. Inside such interval we got values of z, which are inside the $[-0.210401,0.991247]$, as can be seen in Fig. 2. The pairs (x,y) are used as input examples and the *z* as output examples, so we have $e_j = (x_j, y_j, z_j)$ with $j = 1,2,..,225$.

We have run several clustering processes with different numbers of centroids, in particular with 10, 15, 20 and 25 centroids. Starting from those clusters we created the rules following the steps described in 2.2. We applied the two methods described in 3 to those rules. For the method 1 we used a $g_S(t)$ that goes from 0.05 to 0 in 150 Epochs (that never were reached). For method 2 we used the constants $K_c = K_b = K_v = 0.2$. We will show now several graphics of the results. Those results have been obtained using a validation set of 650 data samples different than the training sample set with the rules generated each Epoch.

In the Fig. 1 we can see the differences between both methods. Method 2 is far more effective than method 1 but the computational cost is also higher. In fact, with a learning function $g_S(t)$ that starts in 0.5, the method 1 is able to reach values near to the final value (0.0122) in only 3 epochs. The method 2 reach very good values after the epoch number 30.



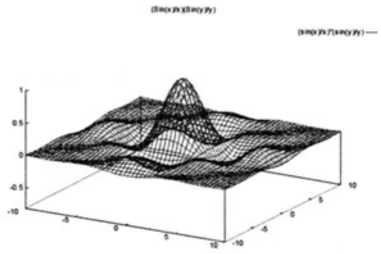
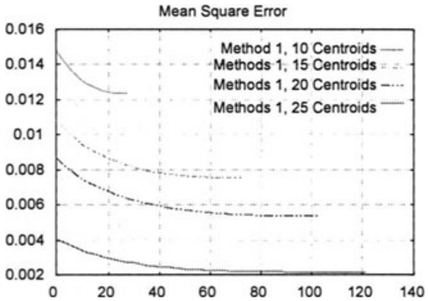


Fig. 1. Comparison of the error obtained by method 1 and method 2 (modifying only the output) with 10 centroids.

Fig. 2. 3-D representation of the function Sinc.



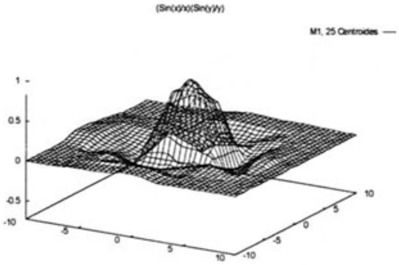


Fig. 3. Comparison of the error obtained by method 1 with regards to the number of centroids.

Fig. 4. Function Sinc approximation using method 1.

In the Fig. 3 we see how, when we add more centroids, we got by far better values of error but at the cost of need more epochs to reach the final values and to add complexity to the computation of them. In the Fig. 5 we see the results using the method 2 with $K_c = K_b = 0$ (so we only modify the rule output), to be able to compare it with method 1 (that only modify the output). We got better results in a few epochs but such epochs are more costly.

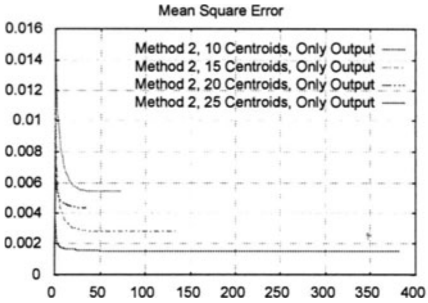


Fig. 5. Comparison of the error obtained by method 2 in regards to the number of centroids.

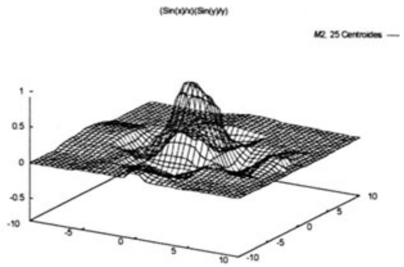


Fig. 6. Function Sinc approximation using method 2.

In the Figures 2, 4 y 6 we appreciate the shape that have the real function and the approximation shape of both methods.

4 Conclusions and Future Trends

In this work we have proposed a fuzzy modeling method consistent in a first approach to the model using clustering techniques that allows us the consideration of a more reduced set of rules, induced from the clusters obtained over the product space of inputs and outputs. Because what we seek in this stage is a first approach, the proposed methods are simple but with a great approximate ability, allowing us to work directly with fuzzy sets in the product space of the input variables.

After having obtained this first model we present two different methods to perform a tune of the fuzzy rules set obtained from the information given by the first clustering. These two methods stand out in a case by its simplicity and in the other case by the great approximation power it reach, at the cost of a greater computational cost.

An aspect to consider once characterized the fuzzy sets of the rules is how can we connect this approach with other similar to them in the literature but that get fuzzy models corresponding to descriptive approaches by a set of predefined labels. In this sense a study is needed about how to connect the rule tune mechanisms with the matrices of weight that shows the certainty we have in each possible combination of labels.

References

1. Ahn, T.;Oh, S.; Woo, K., "Automatic Generation of Fuzzy Rules using the Fuzzy-Neural Networks", Fifth IFSA World Congress, pp. 1181-1185, 1993.
2. Vourimaa, P., "Fuzzy self-organizing map", Fuzzy Sets and Systems, vol. 66, pp. 223-231, 1994
3. Yager, R. R.; Filev, D. P., "Generation of fuzzy rules by mountain Clustering", Tech. Report MII-1318, Iona College, 1993.
4. Gómez Skarmeta, A.F., Delgado, M., Martín, F., "Using Fuzzy Clustering in a Descriptive Fuzzy Modeling Approach", Sixth International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems IPMU'96, July 1996, pp. 563-569, Granada, Spain
5. Yoshinari, Y.; Pedrycz W.; Hirota K., "Construction of Fuzzy Models through Clustering Techniques", Fuzzy Sets and Systems, Vol.54, pp.157-165, 1993
6. Berenji, H.R.; Khedkar, P.S.; "Clustering in Product Space for Fuzzy Inference", Proceedings of the IEEE Int. Conf. on Fuzzy Systems, pp. 1402-1407, 1993
7. Wang, L., Langari. R., Complex Systems Modeling via Fuzzy Logic, IEEE Trans. on Systems, Man and Cybernetics, vol 26, no. 1, pp. 100-105, 1996
8. Ishibuchi, H.; Nozaki, K.; Yamamoto, N.; Tanaka, H., "Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms", Fuzzy Sets and Systems, vol. 65, pp. 237-253, 1994
9. Sun, C.T., "Rule-Base Structure Identification in an Adaptive-Network-Based Fuzzy Inference System", IEEE Transactions on Fuzzy Systems, vol. 2, no. 1, February, pp. 64-73, 1994
10. Delgado, M., Gómez Skarmeta, A.F., Martín F.,"Generating Fuzzy Rules Using Clustering Based Approach", Third European Congress on Fuzzy and Intelligent Technologies and Soft Computing, August. 1995, pp. 810-814, Aachen, Germany
11. Delgado, M., Gómez Skarmeta, A.F., Martín F., "Un Enfoque Aproximativo para la Generación de Reglas mediante Análisis Cluster", V Congreso Español sobre Tecnología y Lógica Fuzzy, Sep. 1995, pp. 43-48, Murcia, España
12. Delgado, M., Gómez Skarmeta, A.F., Vila, A., "On the Use of a Hierarchical Clustering in Fuzzy Modeling", Int. Journal of Approximate Reasoning, vol 14, n°4, 237-259, 1996
13. Kroll, A., "Identification of functional fuzzy models using multidimensional reference fuzzy sets", Fuzzy Sets and Systems, vol. 80, pp. 149-158, 1996
14. Langari. R., Wang, L., "Fuzzy models, modular networks, and hybrid learning", Fuzzy Sets and Systems, vol 79, pp. 141-150, 1996
15. Mizumoto, M. "Method of Fuzzy Inference suitable for Fuzzy Control", *J.Soc. Instrument and Control Engrs*, 58, pp. 959-963, 1989

Chemical Process Fault Diagnosis Using Kernel Retrofitted Fuzzy Genetic Algorithm Based Learner (FGAL) with a Hidden Markov Model

I. Burak Özyurt[1][2], Aydin K. Sunol[1] and Lawrence O. Hall[2]

[1] Department of Chemical Engineering, University of South Florida, Tampa, FL, 33620

[2] Department of Computer Science and Engineering, University of South Florida, Tampa, FL, 33620

Abstract. A hybrid generative-discriminative diagnostic system based on a symbolic learner (FGAL) retrofitted with Gaussian kernel densities for generating instantaneous class probabilities which are further used by a hidden Markov model to estimate the most likely state (fault) given the past evidence is introduced for real time process fault diagnosis. The system allows symbolic knowledge extraction, it is modular and robust. The diagnostic performance of the developed system is shown on a non-isothermal cascade controlled continuously stirred tank reactor (CSTR).

1 Introduction

In the decision making process, a classifier can ignore the relative likelihoods of the candidate classes (posterior probabilities in a statistical sense) or the classifier generates and considers these likelihoods, directly, without modeling the training data or it generates the likelihoods by modeling the training data. The most popular classification systems, such as neural networks and decision trees, are discriminative classifiers belonging to the first two types of classifiers mentioned above. They are only concerned with the boundaries separating the classes and have no distance measure and can generate open decision regions. However, in process fault diagnosis, these approaches have novel class problem, i.e. they will assign a potentially dangerous novel situation to one of the trained classes leaving the novel class unnoticed. One can deal with this problem by modeling the data. A *generative* model [8] directly specifies the manner in which the input data is generated for each class.

Leonard and Kramer ([2]) used a radial basis function network (RBFN) for dynamic chemical process fault diagnosis via a hidden Markov model (HMM) to stochastically model the time component. The methodology introduced here, differs from their approach in the following aspects; a symbolic inductive learning system is used to determine the most descriptive attributes to discriminate each fault class which results in casting from a high dimensional space to a significantly lower dimensional space avoiding the curse dimensionality problem in kernel density estimation and generating an insight about the system behavior in terms of fuzzy rules which RBFNs are not capable of. Besides, using

Baum-Welch method, a way to learn HMM parameters from the process data is introduced which enables the estimation and update of transition probabilities automatically, facilitating the robustness of the diagnostic system.

2 Kernel retrofitted FGAL with hidden Markov model

The real time diagnostic system introduced here has two operating modes. In the learning mode (See Fig. 1a), FGAL is used to generate fuzzy rules from the randomly selected subset of the near quasi-steady state part of normal and faulty process trend data, after FCM based partitioning and conversion of data to linguistic fuzzy variables. Then using the full set of trend data, kernel density estimation is done only for those variables of each class that FGAL generated rules dictate as the most discriminative variables. After that, HMM parameters, a transition probability matrix and initial probabilities are either estimated or learned. For the estimation of parameters, after making the assumptions, that each fault is equally probable and there is no transition from a fault to another, the mean time between failure and knowledge of intermittent fault duration is used ([2], [7]). Using the available database of sequence of state transitions (from normal state to faulty state and vice versa) the parameters can be learned via the Baum-Welch method. These operations are conducted off-line. In the diagnosis mode as depicted in Fig. 1b, the observed process measurements are fed to the system at every sampling time, where for each class, using the scaled product kernel, the conditional density of the observed input data $\mathbf{x}$ given class j $\hat{f}(\mathbf{x}|C_j)$ is generated. The instantaneous posterior probabilities $\hat{p}(C_j|\mathbf{x})$ are calculated using Baye's rule. The instantaneous posterior probabilities are then fed to the hidden Markov model, which decides for each class the probability of the occurrence of that class at the current time given the observed sequence so far. Actually, HMM only takes the previous time step, due to the Markovian assumption, into account.

Although it is probable that, no historical sequence of state transition data is available, during the diagnosis the system can keep track of the instantaneous posterior probabilities and update the transition probability matrix and initial probabilities from time to time in batch-incremental fashion for a spare HMM which is initially identical to the HMM used for diagnosis. In case of significant discrepancy from the original HMM parameters, the original model is replaced by the new learned HMM. To handle vast amounts of sequence data, since a process runs (days, weeks maybe months) in the normal operating mode occasionally displaying abnormal behavior, a compressed data representation scheme is also devised. The components of the system are discussed in the following sections.

2.1 Fuzzy c-means (FCM) based Partitioner

The fuzzy c-means algorithm [1] is a well known clustering algorithm. FCM, unlike the crisp clustering algorithms, such as k-means clustering, allows soft

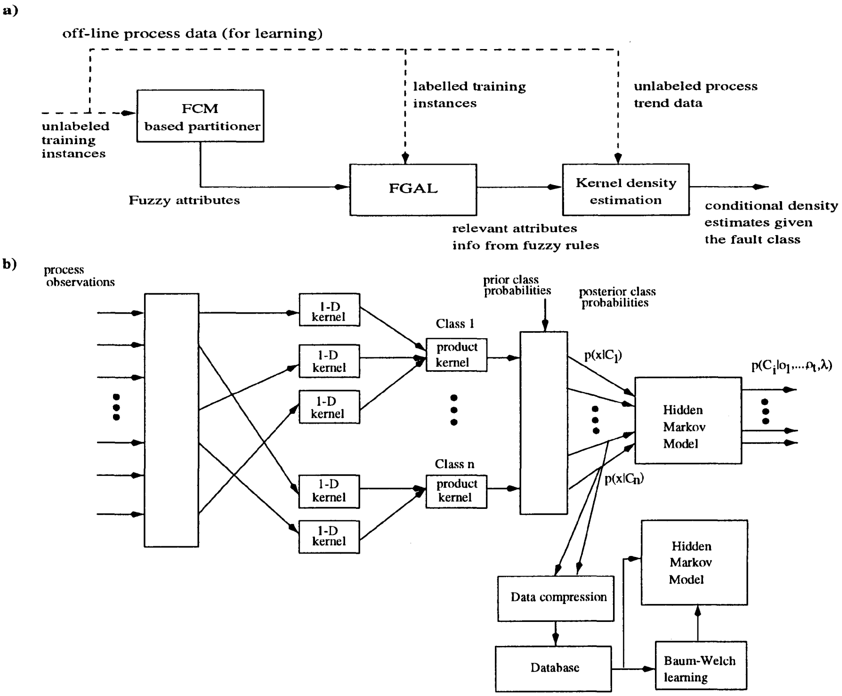

a)
off-line process data (for learning)
unlabeled training instances
FCM based partitioner
labelled training instances
unlabeled process trend data
Fuzzy attributes
FGAL
Kernel density estimation
relevant attributes info from fuzzy rules
conditional density estimates given the fault class
b)
process observations
1-D kernel
1-D kernel
1-D kernel
1-D kernel
1-D kernel
Class 1
product kernel
Class n
product kernel
prior class probabilities
posterior class probabilities
p(x|C1)
p(x|Cn)
Hidden Markov Model
p(Ci|o1,...ot,λ)
Data compression
Database
Hidden Markov Model
Baum-Welch learning

Fig. 1. Kernel retrofitted FGAL with HMM structure. a) off-line mode b) on-line mode.

partitioning, i.e. a pattern can partially belong to more than one cluster, according to its membership value, which is a function of the distance metric selected for clustering. The FCM based partitioner is an interactive, graphic oriented program, which helps the user decide on the symbolic fuzzy attributes used for inductive learning by providing a good estimate of the number of cluster centers and an automated mapping of raw numerical sensor data to linguistic, trapezoidal shaped fuzzy variables. The details of FCM-based partitioner is given in ([3],[4]).

2.2 Fuzzy Genetic Algorithm Based Learner (FGAL)

FGAL uses a genetic algorithm (GA) to search in a fuzzy attribute space for simple and general classification rules. A typical example for a classification rule representation for 3 attributes with 5, 3 and 4 fuzzy set values respectively is

$$\underbrace{\boxed{1}\boxed{0}\boxed{0}\boxed{1}\boxed{0}}_{1}\underbrace{\boxed{\#}\boxed{\#}\boxed{\#}}_{2}\underbrace{\boxed{0}\boxed{0}\boxed{0}\boxed{1}}_{3}\boxed{:}\boxed{2}$$

which is equivalent to

IF (attribute 1 value 1 or value 4) and (attribute 3 value 4) THEN Action (or Category) corresponding to class number 2.

Note that second attribute is denoted as irrelevant for the concept description and dropped by the wildcard operator.

In FGAL, mutation and crossover operators use knowledge about the feature space they are applied in and they avoid searching infeasible territories. Instead of using the traditional non overlapped population (i.e. in every iteration the previous population dies off) an overlapped population using crowding where individuals replace existing strings according to their similarity, is used. Crowding allows FGAL to maintain more than one species (i.e. distinct classification rule describing a portion of a certain class) which can disjuncted to a disjunctive normal form (DNF). Also, it allows the use of very small subpopulations for each class under consideration, which decreases computation time considerably. To guarantee that the very best individuals are not lost from the population by chance, an elitist scheme is used in FGAL.

In evaluation of rules, several learning criteria should be taken into account including complexity, consistency and completeness of a rule. Also the fuzzy nature of the attributes must be considered. Complexity of a rule is usually correlated to the number of tests in the premise part of a rule. The more complex a rule is, the less general it becomes. Completeness, here is viewed as the measure of a rule's domain coverage, i.e. how many instances belonging to a particular class are represented by the rule. A rule's consistency measure describes, how well a rule distinguishes its class from the other classes. The evaluation or fitness function incorporates all the aforementioned criteria. The details of FGAL[3] is given in ([3],[4]).

2.3 Kernel Density Estimation

Density estimation is the construction of an estimate of the probability density function from the observed data which is assumed to be a sample from a unknown probability density function [6]. Kernel density estimation is a non-parametric method where the unknown probability density is estimated using the superimposed local basis functions (kernel functions) as defined by

$$\hat{f}(x) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - X_i)}{h}\right) \ . \tag{1}$$

where $K(\cdot)$ is the kernel function, usually a symmetric probability density function such as Gaussian, satisfying $\int_{-\infty}^{\infty} K(x)dx = 1$, h is the Parzen window width or bandwidth, n is the number of the observed data, X_i is the i^{th} observed data instance. The only parameter to be adjusted in Kernel density estimation is the bandwidth. An inadequate selection of bandwidth results in either too spiky density estimation if the bandwidth selected is too small. If the bandwidth is too large, the resulting density estimate is too obscured, hiding the important

[3] The C++ code for FGAL, FCM-based partitioner, kernel density estimation and HMM calculations with instructions on compiling and using can be downloaded from `http://morden.csee.usf.edu/~ozyurt/progs.html`.

multi-modal characteristics of the underlying density. Since the optimal bandwidth depends on the unknown probability density to be estimated, one cannot analytically solve for it. Therefore, one has to look for other methods for optimal bandwidth selection.

Likelihood cross-validation. The cross-validation method used for optimum kernel bandwidth estimation is based on the maximum likelihood principle. In a classical sense, the maximum likelihood principle considers the best "explanation" of the observed data to be the probability model θ, that maximizes the likelihood function, which maximizes the probability of getting as the result what was actually observed. In density estimation, the probability model θ is to be estimated from the data which will also be used to test the goodness of fit of the statistical model. One can use, leave one out cross-validation method for this purpose. The score function for likelihood cross-validation can be defined as the likelihoods for each data point x_i averaged, i.e.

$$CV(h) = \frac{1}{n} \sum_{i=1}^{n} \log \hat{f}_{-i}(x_i) \ . \tag{2}$$

The score CV is a function of bandwidth h, since the density estimate $\hat{f}$ for a fixed data set is a function of bandwidth and kernel type. The optimum choice of bandwidth h by likelihood cross-validation is then

$$h_{CV} = \arg\max_{h} \left\{ \frac{1}{n} \sum_{i=1}^{n} \log \hat{f}_{-i}(x_i) \right\} \ . \tag{3}$$

The likelihood cross validation method described is for the 1-D case, only. For the multivariable case, the curse of dimensionality problem makes multivariable kernel density estimation inefficient, since the required sample size for a given accuracy grows exponentially with the dimensionality [6]. However, using FGAL, to select the important dimensions of the observations for class discrimination, analogous to Smyth, et. al ([9]) decision tree retrofitting, this dimensionality problem is drastically reduced. Also, utility of the product kernel instead of the full kernel for multidimensional case which requires $d(d+1)/2$ bandwidth parameters to be estimated instead of d parameters for the product kernel, where d is the dimensionality of the problem, also reduces curse of dimensionality.

A two stage 1-D search procedure is used for optimal bandwidth selection using the likelihood cross-validation, for each relevant dimension of each class. In the first stage, a coarse grid search (5 grid points) is applied to zero mean, unit variance scaled n data points within the initial interval of bandwidth search of $[0.25n^{-1/5}\sigma, 1.5n^{-1/5}\sigma]$ to refine the interval of search. In the second stage, Fibonacci search is used for refined search. In the diagnosis, observed data vector $\mathbf{x}$ at sampling time t is used to calculate the conditional density of x given the class j, from the following variant of product Kernel

$$\hat{f}(\mathbf{x}|C_j) = \frac{1}{N_j} \sum_{i=1}^{N_j} \prod_{k \in attr(j)} \frac{1}{h_k} K\left(\frac{x^k - X_i^j}{h_k}\right) \ . \tag{4}$$

where $attr(i)$ denotes the set of fuzzy attributes in the premise part of the best rule for class j and N_j is the number of observations used for the kernel estimate of j^{th} class. The instantaneous posterior probability estimates are calculated using Baye's rule as follows

$$\hat{p}(C_j|\mathbf{x}) = \frac{\hat{f}'(\mathbf{x}|C_j)p(C_j)}{\sum_{i=1}^{N_c}\hat{f}'(\mathbf{x}|C_i)p(C_i)} \quad j = 1, \ldots, N_c \tag{5}$$

$$\hat{f}'(\mathbf{x}|C_j) = \left(\frac{\hat{f}(\mathbf{x}|C_j)}{\prod_{k \in attr(j)} \max_{x_k} \hat{f}(\mathbf{x}|\mathbf{C_j})}\right)^{1/card(attr(j))} \quad . \tag{6}$$

Here N_c is the number of classes and $p(C_j)$ is the prior probability of a class. The scaling of $f(\mathbf{x}|C_j)$ was necessary, since the cardinality of set of attributes $attr$ vary from one class to another, which makes their conditional density estimates incomparable resulting in incorrectly favoring the classes with lower dimensional product kernels. The scaling mechanism in Eq. 6 eliminates this bias.

At every sampling time, calculation of the kernel density estimation involves the superposition of the effect of each training data stored. This has two drawbacks, namely, the need for storage of data used for h_{CV} estimation and the computational load. To facilitate real time operation, instead of calculating the kernel density estimate for every observation from scratch, a simple table lookup with linear interpolation is employed to directly map the observation to the corresponding density estimate. Although this method does not eliminate the data storage problem, it is of $O(1)$. The approximation error introduced is very small and did not pose any significant problem as indicated by the case studies conducted.

2.4 Hidden Markov Models for Chemical Process Fault Diagnosis

Real time monitoring and diagnosis of faults is decomposed into instantaneous estimation of the current state of the process by Kernel retrofitted FGAL and estimation of the sequence of the states by a hidden Markov model(HMM). In standard Markov models, states correspond to deterministically observable events. However, in many real life situations, states are not directly observable, e.g. in a chemical process an occurring fault is only observable through the discrepancy from the normal behavior of noisy sensor measurements. For cases in which the observation is a probabilistic function of state, a ***hidden Markov model*** is defined as a doubly embedded stochastic process with an underlying stochastic process that is not directly observable (hidden) but can be observed only through another set of stochastic processes that produce the sequence of observations [5].

A hidden Markov model is characterized by

- the number of states in the model N (in many applications a physical significance is attached to the states, e.g. for chemical process fault diagnosis states correspond to the faults and normal operation, despite the fact that they are hidden in the observed sensor measurements).

- the state transition probability distribution $\mathbf{A} = \{a_{ij}\}$.
- the observation probability distribution, $\mathbf{B} = \{b_j(\mathbf{o})\} 1 \leq j \leq N$, where $\mathbf{o}$ is observation vector. $b_j(\mathbf{o})$ is defined as $b_j(\mathbf{o}) = P\{\mathbf{o}_t | q_t = j\}$. For the fault diagnosis problem, the observation probability at time t is the instantaneous probability of observing symptoms $\theta(t)$ given that the current state is j^{th} class (fault), which can be stated as $b_j(\mathbf{o}) = P\{\theta(t) | q_t = C_j\}$
- the initial state distribution $\boldsymbol{\pi} = \{\pi_i\}$ where $\pi_i = P\{q_1 = i\}$ $1 \leq i \leq N$.

A HMM is thus, characterized by $\mathbf{A}, \mathbf{B}$ and $\boldsymbol{\Pi}$ or $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\Pi})$. Given an observation sequence $O = (\mathbf{o_1 o_2} \ldots \mathbf{o_T})$ of sensor measurements, the purpose of process fault monitoring and diagnosis is inferring the current state of the process and informing the operator. The probability of the observed sequence (e.g. the probability of occurrence of a fault candidate) given the model λ is calculated using the polynomial time *forward procedure* [5]. In the forward procedure, a forward variable $\alpha_t(i)$ is defined $\alpha_t(i) = P\{\mathbf{o_1 o_2} \ldots \mathbf{o_t}, q_t = i | \lambda$. For fault diagnosis, what is needed at time t is the probability of the i^{th} class (fault) given the observed sequence so far and the HMM. From the definition of forward variable using the definition of conditional probability, it can be calculated as $P\{q_t = i | \mathbf{o_1 o_2} \ldots \mathbf{o_t}, \lambda\} = \frac{\alpha_i(t)}{\sum_{j=1}^{N} \alpha_j(t)}$.

Learning HMM parameters (Baum-Welch method). The HMM parameters $\lambda = (\mathbf{A}, \mathbf{B}, \boldsymbol{\pi})$ depend on the application and they have to be estimated. From these parameters, observation probabilities are estimated from the product kernel density retrofitted FGAL. The other parameters can be estimated via the Baum-Welch iterative method [5], which locally maximizes the likelihood, $P\{O|\lambda\}$ of the HMM by adjusting model parameters λ. The new estimates π_i' and a_{ij}' can be written in terms of forward and backward variables,

$$\begin{aligned} \pi_i' &= \frac{\alpha_0(i)\beta_0(i)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \quad 1 \leq i \leq N \\ a_{ij}' &= \frac{\sum_{t=1}^{T} \alpha_{t-1}(i) a_{ij} . b_j(\mathbf{o_t}) \beta_t(j)}{\sum_{t=1}^{T} \alpha_{t-1}(i) \beta_{t-1}(i)} \quad 1 \leq i, j \leq N \ . \end{aligned} \tag{7}$$

Here, the backward variable $\beta_t(i)$ is defined as $\beta_t(i) = P\{\mathbf{o_{t+1} o_{t+2}} \ldots \mathbf{o_t} | q_T = i, \lambda\}$ and it is calculated using *backward* procedure [5].

For a process, where the sampling time to mean time to failure ratio is small, a vast amount of data have to be stored for the batch Baum-Welch method (it is a batch method, because of the necessary backwards procedure). To facilitate Baum-Welch learning, a compressed data representation scheme is devised. In this method the sequence is automatically segmented, where the segments are represented by a prototype (the mean of the data belonging to that segment). The procedure applied to simulated test sequences has resulted in about a 4:1 compression ratio and the reconstructed sequence is then used to estimate the HMM model parameters via Baum-Welch method resulting in practically the same transition probability matrix as in the original case.

3 Case Study: Non-isothermal Cascade Controlled CSTR

The case study on which the introduced system is tested, is the dynamic simulation of a nonlinear, closed-loop jacketed CSTR with an irrevervisible, exothermic main reaction $A \rightarrow B$, with a consecutive side reaction $B \rightarrow C$, producing unwanted component C. The system has three feedback controllers to control the fluid level in the reactor, the cooling water flow rate and the reactor temperature. Temperature control is accomplished by cascading measurements of reactor temperature and cooling water flow rate to adjust the cooling water flow. The system has multiple interacting feedback loops, nonlinear relationships and shows compensating and inverse response, which makes fault diagnosis challenging. Nine process faults and seven sensor faults are considered. Thirteen process variables are monitored.

The training data is generated by numerical simulation of the cascaded CSTR process for normal and faulty modes with 1% standard deviation from the nominal value via Gaussian noise in observed variables except 1 degree standard deviation Gaussian noise in the temperature measurements, for 12 process hours. A sampling time of 6 minutes is used. To train FGAL, 5 training instances per class are sampled from the near quasi-steady state portion (last one and half hour part) of the trend data. Using 9 rules per class, FGAL is trained for 30 generations. The training time is about 3 minutes on a 486 DX-2 66 microprocessor. The average rule evaluation after 50 runs with different random seeds is 0.7869 with standard deviation 0.0049. The average attribute count is 2.09 per rule with standard deviation 0.11. Hence, FGAL has generated simple, general rules which also appear quite stable in regard to different initializations. Instead of using 13 dimensions, on the average only 2 dimensions are used for the kernel density estimation. Using the relevant dimensions from the general rules learned by FGAL, the kernel density estimation management program, automatically finds the optimal kernel bandwidth using likelihood cross-validation and generates for each relevant dimension of each class a separate interpolator object and saves the lookup table information. For the case studies, the Gaussian kernel $K(t) = \frac{1}{\sqrt{2\pi}} e^{-(1/2)t^2}$ which is common in practice is used. For 36 1-D kernels in total for 17 classes, the optimum bandwidths are estimated automatically and function approximator objects are generated. Total execution time for this operation using 120 data points per kernel is about 5 minutes. The diagonal entries of the transition probability matrix a_{ii} are calculated using the mean time of duration in each state i, $\overline{d}_i$ which can be written as (for unit sampling time) $\overline{d}_i = \sum_{t=1}^{\infty} t p_i(t) = \sum_{t=1}^{\infty} t(a_{ii})^{t-1}(1 - a_{ii}) = \frac{1}{1-a_{ii}}$. The formula for the calculation of a_{ii} is then simply $a_{ii} = 1 - \frac{T}{\overline{d}_i}$, where T is the sampling time. Also assuming the transition from one state to another is equally probable, the full transition probability matrix can be specified. The transition probability matrix corresponding to 100 hours of mean duration time in the normal state and 10 hours of mean duration in any of the faulty states is constructed, in this manner. Thirty test cases per class is generated using the cascade controlled CSTR fault simulator. Using the diagnostic system in on-line mode the test cases are then monitored after moving average filtering. Unlike steady-state fault diagnosis, in dynamic fault diagnosis the fault symptoms develop over time. Hence, as a performance criteria, normalized, time averaged belief in a fault is used. The performance index PI_i for the i^{th} fault is defined as

$$PI_i = \frac{\int_{t_0}^{t_{final}} P\{q_t = i|\mathbf{o}_1\mathbf{o}_2\ldots\mathbf{o}_t,\lambda\}dt}{\sum_{j=1}^{N_f+1}\int_{t_0}^{t_{final}} P\{q_t = j|\mathbf{o}_1\mathbf{o}_2\ldots\mathbf{o}_t,\lambda\}dt} . \tag{8}$$

The performance index ranges between 0 and 1. The closer the performance index of a fault is to one and/or the higher its ratio to other classes is, the better it is distinguished from the other classes. The performance indexes for the three most probable fault candidates with the standard deviations are shown in Fig. 2. The diagnostic system has diagnosed all faults correctly, as the performance indices indicate. The high standard deviations for normal class and fault classes 6, 8, 12 and 14 are due to the conservative, kernel generated decision regions and can be reduced using extra training data for the kernels of those classes.

4 Conclusions

For dynamic systems, kernel retrofitted FGAL with HMM supplies reliable, real time monitoring and diagnosis with symbolic knowledge extraction capability, ease of maintenance due to modularity (a new fault class can be introduced or an existing class can be updated without disturbing the other classes) and HMM parameter learning capability via Baum-Welch method. The diagnostic system minimizes the curse of dimensionality problem of multivariable kernel density estimation using only the relevant features for the detection of a particular fault and avoids novel class problem of discriminative classifiers (e.g. neural nets, decision trees), thereby increasing the reliability of the fault decision.

References

1. Bezdek, J. C. *Pattern Recognition with Fuzzy Objective Function Algorithms.* Plenum, New York (1981).
2. Leonard, J. A. and M. A. Kramer. Diagnosing dynamic faults using modular neural nets. *IEEE Expert*,8 (2), 44-53 (1993).
3. Özyurt, I. Burak. *Chemical Process Fault Diagnosis using pattern recognition and semi-quantitative model based methods.* Ph.D. Dissertation in Engineering Science, University of South Florida (1998).
4. Özyurt, I. Burak and Aydin Sunol. Fuzzy genetic algorithm based inductive learning system (FGALS): a new machine learning approach and application for chemical process fault diagnosis. in *Applications of AI in Engineering XI*, Computational Mechanics Publications (1996).
5. Rabiner, L. and B. H. Juang., *Fundamentals of speech recognition.* Prentice Hall, New Jersey (1993).
6. Silverman, B. *Density estimation for statistics and data analysis.* Chapman and Hall, New York (1986).

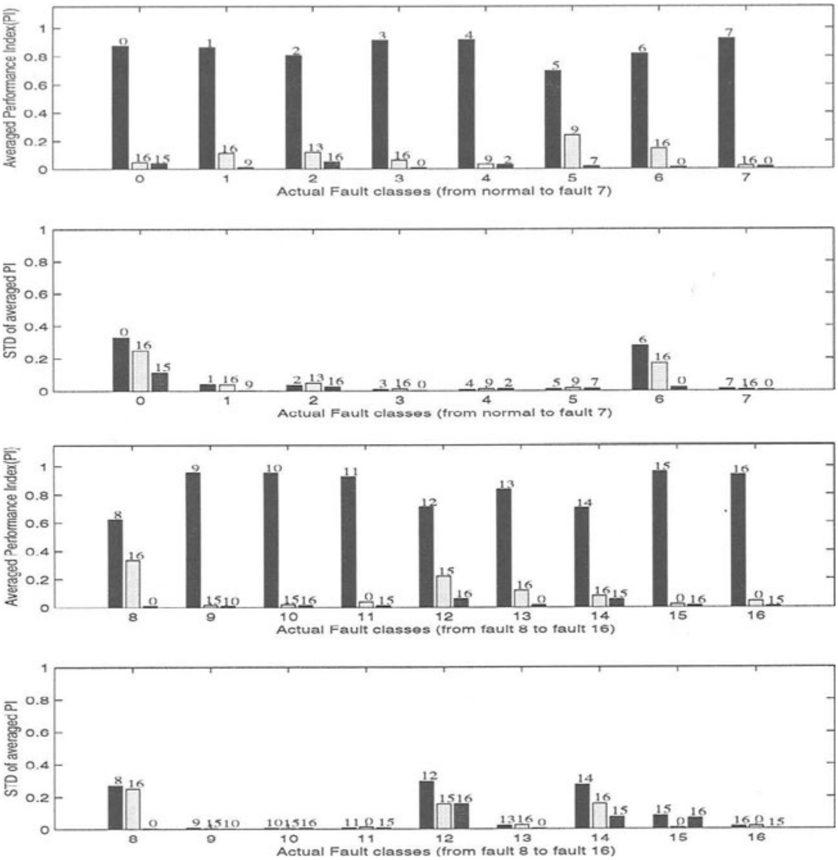


Fig. 2. Diagnosis performance for cascade controlled CSTR

7. Smyth, P. Hidden Markov models for fault detection in dynamic systems. *Pattern Recog.*, **27**, 149-164 (1994).
8. Smyth, P., Probability density estimation and local basis function neural networks. in S. H. Hanson, T. Petsche, M. Kearns and R. Rivest (editors) *Computational learning theory and natural learning systems Volume II: Intersections between theory and experiment* MIT Press, Massachusetts, 233-248 (1994b).
9. Smyth, P, A. Gray and U. M. Fayyad. Retrofitting decision tree classifiers using kernel density estimation. *Proceedings of the 12th International Conference on Machine Learning*, CA (1995).

On Linguistic Approximation with Genetic Programming

Ryszard Kowalczyk

CSIRO Mathematical and Information Sciences
723 Swanston Street, Carlton 3053, Australia
ryszard.kowalczyk@cmis.csiro.au

Abstract. Linguistic approximation is a well known problem in fuzzy set theory to describe an arbitrary fuzzy set in the most appropriate linguistic terms. It involves a search through the space of linguistic descriptions (labels) generated according to a given grammar and vocabulary. The vocabulary and grammar specify a language structure, i.e. allowable and meaningful combinations of primary linguistic terms, linguistic modifiers and linguistic connectives as well as the order of their composition. This structure may become quite complex when a larger vocabulary and more complicated grammatical rules are considered. Therefore linguistic approximation may require search techniques that are able to deal effectively with more complex linguistic structures. The paper presents an approach to linguistic approximation based on an evolutionary search technique, genetic programming. The approach is demonstrated with the results of the initial experiments.

1 Introduction

Linguistic approximation plays an important role in computing with words involving a fusion of natural languages and computation with fuzzy variables. Computing with words is a methodology proposed by Zadeh [12, 13] where the objects of computing are words rather than numbers, with words playing the role of labels of information granules. The foundations of this methodology lay in well known concepts in fuzzy logic such as a linguistic variable and granulation, fuzzy inference, fuzzy restrictions (constraints) and fuzzy constraint propagation (e.g. [9, 10, 11]). In general computing with words involves three main steps [12, 13]. The first step is explicitation of propositions expressed in a natural language, i.e. the representation of the linguistic propositions in their canonical forms. The second step involves reasoning about the propositions (fuzzy restriction propagation) with the use of the rules of inference in fuzzy logic. The third step is a re-translation of induced answers or conclusions into propositions expressed in a natural language. This step involves the use of linguistic approximation, which is the focus of this paper. More specifically, the paper presents an approach to linguistic approximation based on an evolutionary search technique, genetic programming.

Linguistic approximation is a well known problem in fuzzy set theory to describe an arbitrary fuzzy set in the most appropriate (meaningful) linguistic terms. More formally the problem of linguistic approximation consists of associating a linguistic label with the membership distribution of a fuzzy set on the basis of semantic similarity [1, 3, 9]. Many approaches to linguistic approximation have been developed with the use of sets of predefined primary terms (e.g. small, medium, large), linguistic modifiers or hedges (e.g. not, much, very, more or less) and their connectives (e.g. and, or). For example Bonissone [1] has developed a linguistic approximation method based on feature extraction and pattern recognition techniques. A more general approach to linguistic approximation has been proposed by Eshragh and Mamdani [3] which uses a combination of segments of the membership function with well defined characteristics, i.e. S^+ and S^- types of fuzzy subsets. The segments are labeled with the use of linguistic modifiers of the generated primitive terms and the final approximation is a combination of these labels. A similar method of linguistic approximation has been presented in [2] with the use of only linguistic terms entering the inference mechanism of a fuzzy control system.

The process of linguistic approximation involves search through the space of linguistic descriptions (labels) generated according to a given grammar and vocabulary. The vocabulary and grammar specify a language structure, i.e. allowable and meaningful combinations of primary linguistic terms, linguistic modifiers and linguistic connectives as well as the order of their composition. This structure may become quite complex when a larger vocabulary and more complicated grammatical rules are considered. The existing linguistic approximation methods usually use simple search techniques and heuristics that may be inadequate for such problems. Therefore linguistic approximation may require more powerful search techniques in order to deal with more complex linguistic structures.

The problem of finding a linguistic expression composed of primary terms, modifiers and connectives in linguistic approximation has a striking similarity to problems addressed by genetic programming (GP) [7]. GP is an adaptive search technique based on the principles of natural evolution similarly to other evolutionary algorithms such as genetic algorithms [4, 5]. The main distinction is that the structures undergoing adaptation in GP are trees representing symbolic expressions composed of terminals and functions. Due to the closure property each expression has to be valid within the problem domain and some of them have to be sufficient to solve the problem. Although GP has primarily been proposed to generate LISP computer programs [7] and automatically defined functions [8], this approach can also be used to solve problems where solutions have other arbitrary interpretations [6]. It also seems that the principles of GP can be used in linguistic approximation to advantage.

The paper presents and exemplifies an approach to linguistic approximation based on genetic programming. After a brief overview of linguistic approximation in section 2, some aspects of using genetic programming in linguistic approximation are presented in section 3. Two examples of linguistic approximation with the use of genetic programming are illustrated in section 4. Finally, the concluding remarks are presented in section 5.

2 Linguistic approximation

The problem of linguistic approximation can be defined as mapping from a set S of fuzzy subsets in a universe of discourse U, into a set of labels L, which are generated according to a grammar G and a vocabulary V [3]. A solution of linguistic approximation is a linguistic description (label) LA composed of linguistic primary terms T, linguistic modifiers M and linguistic connectives C such that it is most suitable (meaningful) to describe a given fuzzy set. The quality of linguistic approximation is usually a trade-off between an error of the approximation and a structural complexity of the description.

Let us consider a simple problem of linguistic approximation illustrated in Fig. 1.

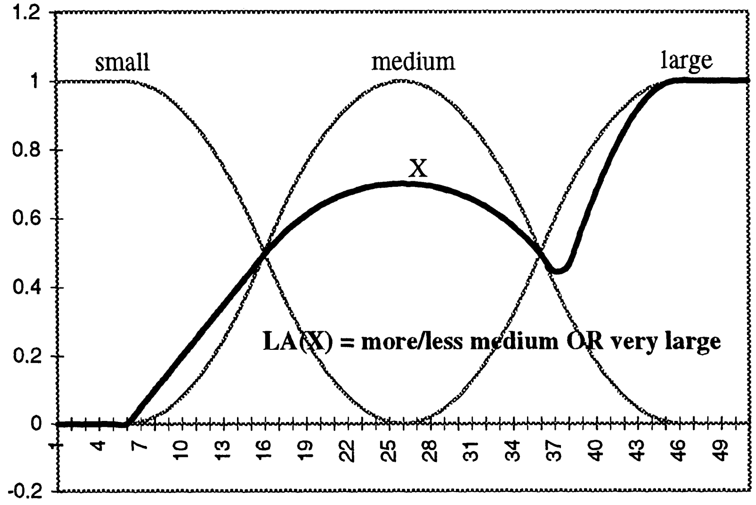


Fig. 1. An example of linguistic approximation

The problem is to assign a linguistic label to a fuzzy set X with the use of the following elements:

- a set of primary terms T = {small, medium, large}
- a set of linguistic modifiers M = {not, very, more/less, indeed, above, below}
- a set of connectives C = {and, or}.

The primary terms and modifiers correspond to the fuzzy subsets in Fig. 1 and Fig. 2, respectively. The definitions of the linguistic modifiers and connectives are listed in Table 1.

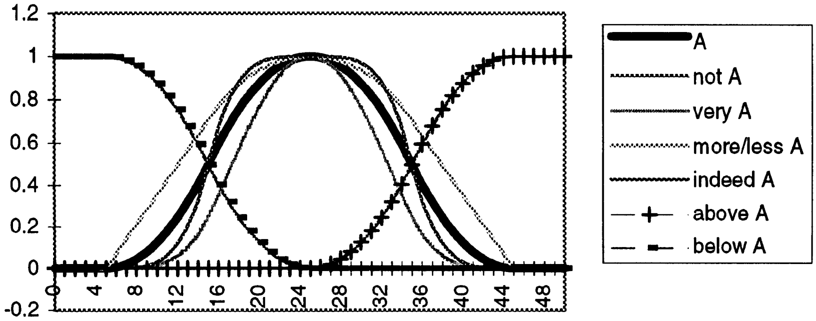


Fig. 2. An example of linguistic modifiers

Table 1. Examples of typical linguistic modifiers and connectives [3]

Linguistic modifier M(X)	membership function $\mu_{M(X)}(x)$
not	$\mu_{not\,X}(x)=1-\mu_X(x)$
very	$\mu_{very\,X}(x)=\mu_X^2(x)$
more or less	$\mu_{more/less\,X}(x)=\mu_X^{1/2}(x)$
indeed	$\mu_{indeed\,X}(x)=\begin{cases}2\mu_X^2(x) & for\ 0\le\mu_X(x)\le 0.5\\ 1-2(1-\mu_X(x))^2 & otherwise\end{cases}$
above	$\mu_{above\,X}(x)=\begin{cases}1-\mu_X(x) & for\ x\ge x_{\max}\\ 0 & otherwise\end{cases}$
below	$\mu_{below\,X}(x)=\begin{cases}1-\mu_X(x) & for\ x\le x_{\min}\\ 0 & otherwise\end{cases}$
and	$\mu_{X_1\,and\,X_2}(x)=\min(\mu_{X_1}(x),\mu_{X_2}(x))$
or	$\mu_{X_1\,or\,X_2}(x)=\max(\mu_{X_1}(x),\mu_{X_2}(x))$

In the considered example the fuzzy set X can be linguistically approximated with a label LA(X)="more/less medium or very large" which can also be written as LA(X) = "more/less(medium) or very(big)" or LA(X) = "or(more/less(medium), very(big))".

It should be noted that even in this simple example the linguistic structure of a label may be quite complex (e.g. "indeed not very very small or more less large"). A larger vocabulary may increase the search space substantially. The generation and search through the search space may be additionally complicated by grammar imposed on the structure of the meaningful linguistic labels (e.g. "indeed very not very small" may not be grammatical). A search technique used in linguistic approximation should be able to take these factors into account.

3 Linguistic approximation with genetic programming

Genetic programming operates similarly to other evolutionary algorithms that provide an adaptive method of search using the principles of evolutionary simulation [4, 5, 7]. A basic behavior of GP is provided by applying evaluation, selection, crossover and mutation on populations of chromosomes representing prospective solutions in the search space. The solutions are tree structures composed of symbolic expressions used in the problem representation. After generating an initial population, GP iteratively applies those four basic steps to generate subsequent populations of evolving chromosomes until a termination condition is satisfied (e.g. the optimal or satisfactory solution is found according to some evaluation criteria or a given number of iterations has been performed). Evaluation and selection ensure that only the most prospective in the terms of fitness chromosomes in a population survive. The objective of selection is to form new populations of chromosomes with higher average fitness. These chromosomes are used next in reproduction using crossover and mutation which introduce new individuals to the population with a hope that they represent better solutions.

The structures undergoing adaptation in GP traditionally represent hierarchical computer programs based on LISP-like symbolic expressions [7, 8]. The size, shape and structure of the symbolic expressions are left unspecified and are found by using the genetic programming operators. Solving a problem therefore becomes a search through all the possible combinations of predefined symbolic expressions in order to find one that represents (solves) the problem according to given criteria. These symbolic expressions are divided into two sets, i.e. a function set and a terminal set. If an expression requires arguments it is placed in the function set, otherwise it is placed within the terminal set. Each individual expression must have the property of closure, i.e. it must be valid within the problem domain, and some combinations must be sufficient to solve the problem. In the context of linguistic approximation the terminal set can consist of the primary terms and the function set can consist of the linguistic modifiers and connectives. Therefore the structures undergoing adaptation are linguistic descriptions composed of the primary terms, modifiers and connectives. Examples of a tree representation of a linguistic description in GP are shown in Fig. 3. The property of closure in GP corresponds to the grammar and the vocabulary in linguistic approximation. The objective of GP applied to linguistic approximation is to find a linguistic structure that approximates a given fuzzy set according to some evaluation criteria (e.g. distance measure between a fuzzy set and its approximation). GP allows also one to take into account the complexity of the solution by giving each solution a structural complexity. The criteria are used to guide the evolution process during evaluation and selection of prospective linguistic expressions and their reproduction with crossover and mutation

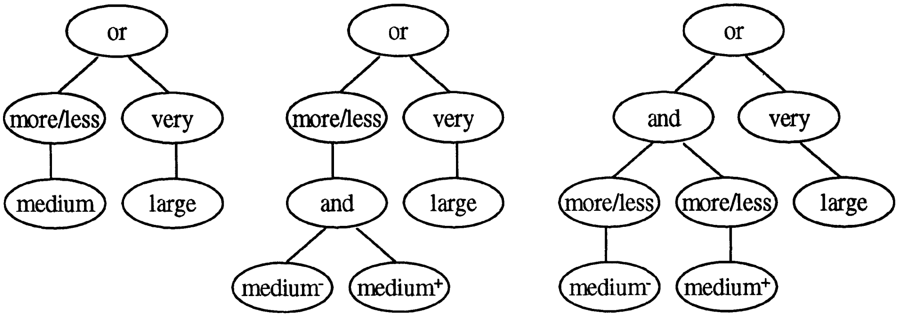


Fig. 3. Tree representation of a linguistic description in GP

Crossover is a reproduction mechanism in evolutionary algorithms creating new chromosomes from the existing ones with a hope that they represent better solutions. Simple crossover in GP selects two structures from the population and selects one point on each. Each sub-tree from this point is swapped with the other. The closure property ensures that the new linguistic structures are still grammatical expressions within the language. An example of crossover of linguistic labels is shown in Fig. 4.

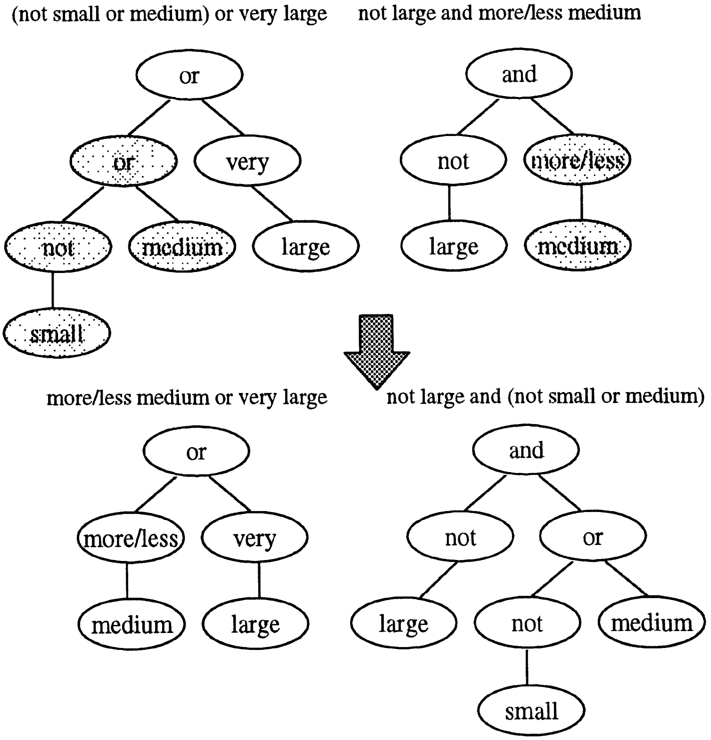

(not small or medium) or very large
not large and more/less medium
or
or
very
not
medium
large
small
and
not
more/less
large
medium
more/less medium or very large
not large and (not small or medium)
or
more/less
very
medium
large
and
not
or
large
not
medium
small

Fig. 4. Crossover of linguistic descriptions in GP

Mutation changes individual chromosomes to provide additional variability to the reproduction process. A basic mutation in GP comprises expressions within the structure being swapped with other expressions within certain constraints. Any terminal can be swapped with any other terminal, but functions can only be swapped with other functions with the same number of arguments. In the context of linguistic approximation it means that the swapping can only be done within sets of the primary terms, linguistic modifiers and linguistic connectives. The closure property again ensures that the new linguistic structure is still valid within the language. An example of mutation of a linguistic label is illustrated in Fig. 5.

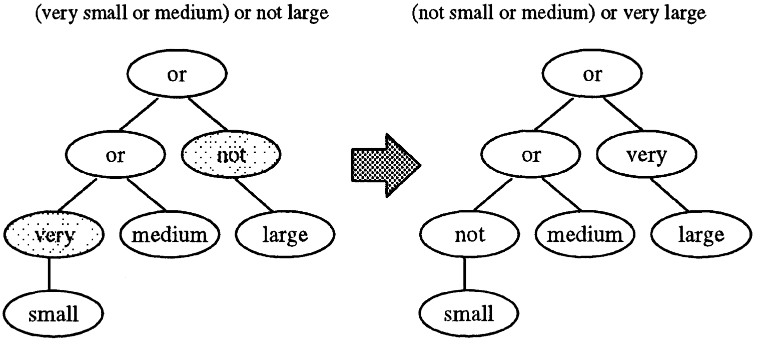

(very small or medium) or not large
(not small or medium) or very large
or
or
not
very
medium
large
small
or
or
very
not
medium
large
small

Fig. 5. Mutation of a linguistic description in GP

4 Experiments

To test the feasibility of the presented approach to linguistic approximation with GP a number of simulated experiments have been conducted. For example two test problems were adapted from Eshragh and Mamdani [3] where they were approximated with the system LAM5. The first problem takes as its input a set of three primary terms: low, medium and high (see Fig. 6), six modifiers: not, very (much), indeed, more or less (more/less), above and below, and two connectives: and, or. The results of the experiment with GP are presented in Table 2 together with the results reported by Eshragh and Mamdani [3].

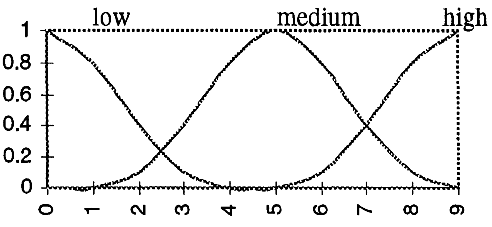


Fig. 6. Definitions of primary sets

Table 2. Results of experiments

Input*	LA with LAM5 [3]	LA with GP
high	high	high
above low	not low	not low
medium	medium	medium
not medium	not medium	not medium
above medium and not high	above medium and not high	above medium and not high
medium or high	medium or high	medium or high
below high	not high	not high
below high and above low	not low and not high	not low and not high
not below high or not above low	low or high	low or high
medium and high	medium and high	*above medium and not high*
medium or not below medium	below medium	*above medium*
very low or low	low	low
low or medium or high	low or medium or high	low or medium or high

The second problem relates to a set of seven primary subsets: negative big NB, negative medium NM, negative small NS, zero ZO, positive small PS, positive medium PM and positive big PB as defined in Fig. 7. The results of the experiment are presented in Table 3.

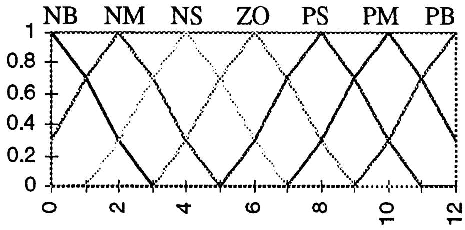


Fig. 7. Definitions of primary sets

Table 3. Results of experiments

Input*	LA with LAM5 [3]	LA with GP
NB or NM	NB or NM	NB or NM
not NB and not NM	not NB and not NM	not NB and not NM
not above ZO and not below ZO	ZO	ZO
not above ZO and not below PS	ZO and PS	*not above ZO and not below PS*
NB or ZO or PS	NB or ZO or PS	NB or ZO or PS

[*] In Eshragh and Mamdani's experiments the input sentences were translated into numerical representation with the use of a program TRAM [3]. In our experiments the numerical input was considered directly according to the definition in Fig. 5 and Table 1.

It can be observed that GP was able to provide very good linguistic approximation for the above test problems. It should be noted that in some cases the results of linguistic approximation with GP (italicized in the Tables 2 and 3) were more grammatical than LAM5's results thanks to the ability of GP to take into account numerous grammatical rules related to the closure property.

5. Conclusion

Linguistic approximation is an important task in the methodology of computing with words that retranslates induced fuzzy conclusions into propositions in natural language. The paper has presented and exemplified an approach to linguistic approximation based on the principles of genetic programming. The initial experiments have indicated that GP is a promising tool in linguistic approximation and more generally it may be useful in computing with words. GP is able to provide meaningful and intuitive linguistic labels taking into account potentially complicated grammar and large vocabulary. Linguistic approximation can be performed with GP according to different criteria such as quality of matching, structural complexity, using preferred vocabulary and grammatical rules. In addition because GP is an adaptive technique it is able to provide a current expression at any time of the evolution process. It may allow one to obtain a "rough" linguistic description quickly and improve it (e.g. make it more precise or more compact) if required.

It should be noted that although it seems that GP is a promising technique in linguistic approximation further research is still required. For example the properties of closure and sufficiency need more detailed investigation in the context of handling complicated vocabulary and grammar as well as the use of other genetic operators and their impact on performance of linguistic approximation with GP.

References

1. Bonissone P.: A fuzzy set based linguistic approach: Theory and applications. In: M.M. Gupta and E. Sanchez (eds.): Approximate Reasoning in Decision Analysis, North-Holland (1982) 329-339.
2. Dvorak A.: On Linguistic Approximation in the Frame of LFLC. Proc. Seventh IFSA World Congress, Prague (1997) 413-417.
3. Eshragh F. and Mamdani H.: A general approach to linguistic approximation. In: E.H. Mamdani and B.R. Gaines (eds.) Fuzzy Reasoning and its Applications, Academic Press (1981) 169-187.
4. Goldberg D. E.: Genetic algorithms in search, optimization, and machine learning. Addison-Wesley (1989).
5. Holland J.: Adaptation in Natural and Artificial Systems. University of Michigan Press (1975).

6. Kinnear K. E. (ed): Advances in Genetic Programming (1994).
7. Koza J.: Genetic Programming, MIT Press (1992).
8. Koza J.: Genetic Programming II: Automatic Discovery of Reusable Programs (1994).
9. Zadeh L. A.: Outline of a new approach to the analysis of complex systems and decision processes, IEEE Trans. Man. Cybernetics, No.3 (1973) 28-44.
10. Zadeh L. A.: Calculus of fuzzy restrictions. In: L.A. Zadeh. K.S. Fu, M. Shimura, (eds.): Fuzzy Sets and Their Applications to Cognitive and Decision Processes,. New York: Academic (1975) 1-39.
11. Zadeh L. A.: PRUF - A meaning representation language for natural languages. Int. J. Man-Mach. Studies, vol. 10 (1978) 395-460.
12. Zadeh L. A.: Soft Computing and Fuzzy Logic. IEEE Software, 11(6) Nov (1994) 48-56.
13. Zadeh L. A.: Fuzzy Logic = Computing with Words. IEEE Transactions on Fuzzy Systems, vol. 4, no. 2, May (1996) 103-111.

A New Solution Methodology for Fuzzy Relation Equations

Spyros G. Tzafestas[1] and Giorgos B. Stamou[2]

[1] Intelligent Robotics and Automation Laboratory,
Department of Electrical and Computer
Engineering,
National Technical University of Athens,
Zographou 15773, Athens, Greece.
e-mail: tzafesta@softlab.ece.ntua.gr
[2] Intelligent Robotics and Automation Laboratory,
Department of Electrical and Computer
Engineering,
National Technical University of Athens,
e-mail: gstam@softlab.ece.ntua.gr

Abstract. In this paper a general methodology for studying and solving fuzzy relation equations based on sup-t composition, where t is any continuous triangular norm, is proposed. To this end the concept of the "solution matrices" is introduced, as a way of representing the process information required for the resolution. Using this concept, the solution existence of a fuzzy relation equation is first examined. When the relation equation has no solution, the reasons for this lack of solvability are found. Otherwise, the *solution set* is determined.

1 Introduction-Problem Statement

The resolution problem of *fuzzy relation equations* (FREs) is one of the most important and widely studied problems, in the field of fuzzy sets and fuzzy systems. It has been established ([3], (pp. 321-323)) that the majority of fuzzy inference systems can be implemented using FREs. Furthermore, it is well known that this implementation is feasible and convenient for many reasons ([2], [5], [7]). Different kinds of FREs have been studied ([2], [4], [6]). Although the problem for the sup-min composition case has been solved, till now no general method has been developed, for determining the solution set of a sup-t FRE, where t is an arbitrary continuous t-norm. For this type of equations, methods for determining only the maximal solution (and *not* the minimal solutions) have been proposed ([2], [3] (pp.162-164)). This is the subject of the present paper. The concepts of the "solution matrices" and the "mean solution" are introduced, which enable one to combine and generalize the ideas proposed in the above papers, and to provide a complete methodology for solving sup-t FREs (finding the solution set), where t is a continuous t-norm. The use of general sup-t compositions for fuzzy inference is lately enforced, since it is proved ([1]) that the outputs of fuzzy systems are sensitive with respect to the choise of the t-norm.

Let X, Y, Z be discrete crisp sets with cardinalities n, m and k, respectively, and $A(X,Y)$, $R(Y,Z)$, $B(X,Z)$ be three binary fuzzy relations constraining its other with the relationship

$$A \circ^t R = B \tag{1}$$

where $\circ^t$ is the well known sup-t composition (t is a t-norm). Equation (1) can be written in the matrix form

$$\mathbf{A} \circ^t \mathbf{R} = \mathbf{B} \tag{2}$$

where $\mathbf{A}_{n\times m}$, $\mathbf{R}_{m\times k}$, $\mathbf{B}_{n\times k}$ are the matrix representations of A, R, B, respectively. Equation (2) is the typical form of a FRE, for which the problem that arise is its resolution for **R,** when **A** and **B** are given. (2) is actually a set of k more simple fuzzy relation equations that can be solved independently and so it suffices to consider only the equation:

$$\mathbf{A} \circ^t \mathbf{r} = \mathbf{b} \tag{3}$$

where $\mathbf{r}_{m\times 1}$ and $\mathbf{b}_{n\times 1}$ are column vectors of $\mathbf{R}$ and $\mathbf{B}$, respectively. Clearly, (3) is simply a system of n equations of the form:

$$\mathbf{a} \circ^t \mathbf{r} = b \tag{4}$$

where **a** is a row vector of **A**. Now, one can easily see that (3) has a solution for **r** if and only if (iff) all the n equations of the form of (4) have at least one common solution for **r**. Let $S(\mathbf{A},\mathbf{b})$ be the solution set of (3), i.e. $S(\mathbf{A},\mathbf{b})=\{\mathbf{r}|\mathbf{A}\circ^t \mathbf{r}=\mathbf{b}\}$. It is well known ([2], pp. 157, 162) that if $S(\mathbf{A},\mathbf{b})\neq\varnothing$, then it contains a unique maximal solution $\hat{\mathbf{r}}$ and may contain several minimal solutions $\breve{\mathbf{r}}$. In this case, the solution set is the union of all the lattices $[\breve{\mathbf{r}},\hat{\mathbf{r}}]$ between each minimal and the maximum solution. In figure 1, a view of the solution set is illustrated.

We define the mean solution, as the minimal element of the intersection of the lattices $[\breve{\mathbf{r}},\hat{\mathbf{r}}]$. Obviously, the mean solution always exists (since $\hat{\mathbf{r}}$ always belongs to the intersection of the lattices $[\breve{\mathbf{r}},\hat{\mathbf{r}}]$) and it is unique. In the sequel, we solve equation (3), when t is an arbitrary continuous t-norm, giving a general method for obtaining the maximum, the mean and all the minimal solutions. Then, the maximum, the mean and the minimum solutions of (2) are the $m\times k$ matrices $[\hat{\mathbf{r}}_1\, \hat{\mathbf{r}}_2\, \cdots\, \hat{\mathbf{r}}_k]$, $[\bar{\mathbf{r}}_1\, \bar{\mathbf{r}}_2\, \cdots\, \bar{\mathbf{r}}_k]$ and $[\breve{\mathbf{r}}_1\, \breve{\mathbf{r}}_2\, \cdots\, \breve{\mathbf{r}}_k]$, where $\hat{\mathbf{r}}_i$ ($i=1,2,...k$) is the maximum solution, $\bar{\mathbf{r}}_i$ ($i=1,2,...k$) is the mean solution and $\breve{S}_i$ ($\breve{\mathbf{r}}_i \in \breve{S}_i$ ($i=1,2,...,k$)) is the minimal solutions set of the i-th form (3) equation.

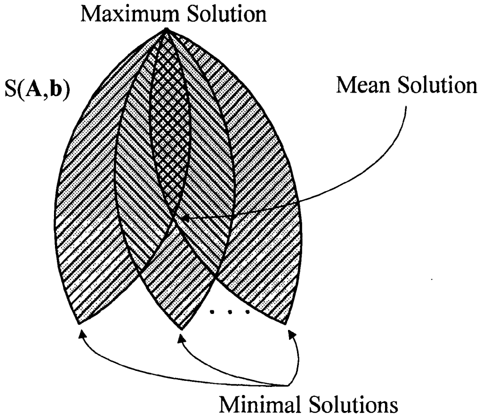


Figure 1: The solution set of the fuzzy relation equation

2 Solution Matrices

A function t:$[0,1]\times[0,1]\to[0,1]$ is a t-norm iff $\forall a,b,d\in[0,1]$ it satisfies the next four axioms (axiomatic skeleton for t-norms):

Axiom 1: $t(a,1)=a$ and $t(a,0)=0$ (boundary condition)

Axiom 2: $b\le d$ implies $t(a,b)\le t(a,d)$ (monotonicity)

Axiom 3: $t(a,b)=t(b,a)$ (commutativity)

Axiom 4: $t(a,t(b,d))=t(t(a,b),d)$ (associativity)

A t-norm t is called *Archemedean* iff

Axiom 5. t is a continuous function

Axiom 6. $t(a,a)<a,\ \forall a\in(0,1)$ (subidempotency)

The class of t-norms has been widely studied by many researchers and it is proved in [3] (p. 63) that $\min(a,b)$ is the only idempotent t-norm. On the other hand, for any t-norm t it is true that ([3], p. 65) $t(a,b)\le\min(a,b)$. Thus, we can say that the only continuous t-norm which is not Archemedean is the $\min(a,b)$. This is a very interesting result, since it suggests a separate study of (3) when t is an Archemedean t-norm and when t is the $\min(a,b)$.

Let $a\,\widehat{\otimes}^t\,b$ and $a\,\breve{\otimes}^t\,b$ denote the maximal and minimal solution, respectively, of the equation $t(a,x)=b$ (if they exist), i.e.

$$a\,\widehat{\otimes}^t\,b=\sup\{x\in[0,1]\mid t(a,x)=b\},\quad a\,\breve{\otimes}^t\,b=\inf\{x\in[0,1]\mid t(a,x)=b\}$$

If the solution of $t(a,x)=b$ is unique, then it is denoted by $a\otimes^t b$.

Then, the useful operation $\hat{\omega}_t(a,b) = \sup\{x \in [0,1] \mid t(a,x) \le b\}$ defined in [3], takes the form $\hat{\omega}_t(a,b) = \begin{cases} 1 & a < b \\ a \hat{\otimes}^t b & a \ge b \end{cases}$ for any $a,b \in [0,1]$.

Furthermore, for every $a,b \in [0,1]$, we define the operation

$$\breve{\omega}_t(a,b) = \begin{cases} 0 & a < b \\ a \breve{\otimes}^t b & a \ge b \end{cases}.$$

In order to fully understand the underlying mechanism of FREs and find the solution set, we introduce the concept of the *solution matrices*.

Definition 1. Let t be a continuous t-norm, and $\mathbf{A} \circ^t \mathbf{r} = \mathbf{b}$ be a FRE of the form (3). The matrix $\Gamma_{m \times n}$ is the *solution matrix* (SM) of the FRE, where:

$$\Gamma_{ij} = \breve{\omega}_t(A_{ji}, b_j), \ \forall\, i \in N_m, j \in N_n .$$

The matrix $\hat{\Gamma}$ is the *maximal solution matrix* (max-SM) of the FRE, where:

$$\hat{\Gamma}_{ij} = \hat{\omega}_t(A_{ji}, b_j), \ \forall\, i \in N_m, j \in N_n .$$

The matrix $\breve{\Gamma}$ is the *minimal solution matrix* (min-SM) of the FRE, where:

$$\breve{\Gamma}_{ij} = \breve{\omega}_{\min}\left(\inf_{k \in N_n} \hat{\Gamma}_{ik}, \Gamma_{ij} \right), \ \forall\, i \in N_m, j \in N_n .$$

Obviously, if $S(\mathbf{A},\mathbf{b}) \ne \varnothing$ it is $\hat{r}_i = \inf_{k \in N_n} \hat{\Gamma}_{ik} = \inf \hat{\mathbf{\Gamma}}$, where $\hat{\mathbf{r}}$ is the maximum solution defined in [3] (p.162). Moreover, it is proved in [8] that if $S(\mathbf{A},\mathbf{b}) \ne \varnothing$, then $\bar{\mathbf{r}} = \sup \breve{\mathbf{\Gamma}}$.

3 Solution Existence

The existence of the solution of (3) is one of the most important problems of FREs. In [3] (pp. 146-148) some useful properties of $\hat{\omega}_t$ are established, which lead to the following proposition.

Proposition 1. If $S(\mathbf{A},\mathbf{b}) \ne \varnothing$, then $\hat{\mathbf{r}} = \mathbf{A}^{-1} \circ^{\hat{\omega}_t} \mathbf{b}$ is the greatest member of $S(\mathbf{A},\mathbf{b})$, where $\circ^{\hat{\omega}_t}$ is the $\inf - \hat{\omega}_t$ composition.

Proposition 1 provides a first way of testifying whether $S(\mathbf{A},\mathbf{b}) \ne \varnothing$: if $\mathbf{A} \circ^t \hat{\mathbf{r}} = \mathbf{b}$ then $S(\mathbf{A},\mathbf{b}) \ne \varnothing$.

A more intensive way for testing the existence of the solution is based on the direct comparison of the elements of the matrices **A** and **b**. For this aim, we provide the following Lemmas (their proofs are beyond the scope of this paper; they are included in [8]).

Lemma 1. We have $S(\mathbf{a},b)\neq\varnothing$ iff there exists $j\in N_m$ such that $a_j\geq b$.
Corollary 1. If there exists $j\in N_m$ such that $a_j=1$, then $S(\mathbf{a},b)\neq\varnothing$.

Lemma 2. Let $\mathbf{A}\circ^t\mathbf{r}=\mathbf{b}$ be FRE of the form (3) with $m\geq n$. If for any $i\in N_n$ there exists $j\in N_m$ such that $a_{ij}\geq b_i$ and $a_{kj}\leq b_k$, $\forall k\in N_m-\{i\}$, then $S(\mathbf{A},\mathbf{b})\neq\varnothing$.

Corollary 2. Let $\mathbf{A}\circ^t\mathbf{r}=\mathbf{b}$ be FRE of the form (3) with $m\geq n$. If for any $i\in N_n$ there exists $j\in N_m$ such that $a_{ij}=1$ and $a_{kj}=0,\ \forall k\in N_m-\{i\}$, then $S(\mathbf{A},\mathbf{b})\neq\varnothing$.

The solution matrices are helpful tools for testifying the existense of the solution of a FRE. We now provide the following Lemmas (their proofs are complicated and therefore, we omit them in this limited presentation; they are included in [8]).

Lemma 3. Let t be a continuous t-norm and $\mathbf{A}\circ^t\mathbf{r}=\mathbf{b}$ be a FRE of the form (3).
If $S(\mathbf{A}_{j\bullet},b_j)\neq\varnothing$ for any $j\in N_n$, then:

$$\overset{\frown}{\mathbf{\Gamma}}_{\bullet j}\in S(\mathbf{A}_{j\bullet},b_j),\ \forall j\in N_n$$
$$\mathbf{\Gamma}_{\bullet j}\in S(\mathbf{A}_{j\bullet},b_j),\ \forall j\in N_n$$

where $\breve{\mathbf{\Gamma}}$ and $\mathbf{\Gamma}$ is the min-SM and the SM of the equation, respectively.

Lemma 4. Let $\mathbf{A}\circ^t\mathbf{r}=\mathbf{b}$ be a FRE of the form (4), where t is a continuous t-norm. The following propositions are equivalent:
i) $S(\mathbf{A},\mathbf{b})=\varnothing$.
ii) there exists $j\in N_n$ such that $\breve{\mathbf{\Gamma}}_{\bullet j}=\mathbf{0}$ and $b_j\neq 0$.

Lemma 3 and Lemma 4 give a clear understanding of the solution matrices and useful criteria for the solution existence of the FRE:

i) It is $S(\mathbf{A}_{j\bullet},b_j)\neq\varnothing$, $\forall j\in N_n$ iff $\mathbf{\Gamma}_{\bullet j}=\mathbf{0}$ implies that $b_j=0$. (if there exists $j\in N_n$ such that $\mathbf{\Gamma}_{\bullet j}=\mathbf{0}$ (with $b_j\neq 0$), then the j-th equation of the FRE has no solution, and consequently, the relation equation has no solution.

ii) If $\forall j \in N_n$: $\Gamma_{\bullet j} = \mathbf{0}$ only when $b_j = 0$ and there exists $j \in N_n$ such that $\breve{\Gamma}_{\bullet j} = \mathbf{0}$ (and $b_j \neq 0$), then although all the form (4) FREs of the form (3) FRE have a solution, the form (3) FRE has no solution, since the j-th equation is in contradiction with at least one of the others.

iii) $S(\mathbf{A},\mathbf{b}) \neq \varnothing$ iff for any $j \in N_n$: $\breve{\Gamma}_{\bullet j} = \mathbf{0}$ only when $b_j = 0$.

4 The Proposed Method of Solving FREs

In this section, we first present a general method for finding the set of the minimal solutions. Using this method, the *entire* set of the minimal solutions is determined only when t is an *Archemedean* t-norm. If t is the *minimum* t-norm, a set that *contains* the minimal solutions (and may also contains non-minimal solutions) is determined. Such a type of set is determined with all the well known methods for the resolution of sup-min FRE ([2], [4], [6] etc.).

The problem of finding the minimal solutions can be simplified by reducing the equation that has to be solved. Such reductions which are useful from the practical point of view are the following.

Reduction 1. For any $j \in N_m$ such that $\hat{r}_j = 0$, it is, obviously, $\breve{r}_j = 0$, for any $\breve{\mathbf{r}} \in \breve{S}(\mathbf{A},\mathbf{b})$, and consequently one can eliminate the j-th element of $\mathbf{r}$ as well as the j-th column of $\mathbf{A}$ in the whole process of determining the minimal solutions. The final solution of the FRE results from the solution of the reduced equation by extending it, with the insertion of zeros at the locations that were eliminated.

Reduction 2. For any $j \in N_n$ such that $b_j = 0$, one can eliminate the j-th row of $\mathbf{A}$ as well as the j-th element of $\mathbf{b}$. Then, the solution of the FRE is obviously the same as the solution of the reduced equation. (for any $\mathbf{r} \leq \hat{\mathbf{r}}$ the equation $\mathbf{A}_{j\bullet} \circ^t \mathbf{r} = 0$ is satisfied, since $\mathbf{A}_{j\bullet} \circ^t \hat{\mathbf{r}} = 0$).

For the rest of the paper, when we say that a FRE of the form (3) is in the "reduced form" we mean that $b_j \neq 0$, $\forall j \in N_n$.

We now provide the reader with the determination of some special sets used for the determination of the minimal solutions set. Let $\mathbf{X}_{m \times n}$ be a matrix, with $X_{ij} \in [0,1]$ and $\mathbf{X}_{\bullet j} \neq \mathbf{0}$, $\forall j \in N_n$. The following notations are introduced:

$$\Im_j(\mathbf{X}) = \left\{ i \in N_m | \, X_{ij} \neq 0 \right\}$$

$$\Im(\mathbf{X}) = \left\{\left\{J_j,\ \forall j \in N_n\right\},\ \forall J \in \underset{k\in N_n}{\times} \Im_k(\mathbf{X})\right\}$$

$$\breve{\Im}(\mathbf{X}) = \left\{J \in \Im(\mathbf{X})|\ \forall K \in \Im(\mathbf{X}) - \{J\} \text{ it is } K \not\subset J\right\}$$

$$M(\mathbf{r}_{m\times 1},\mathbf{X}) = \left\{\mathbf{s}_{m\times 1}(J)|\ s_i = \begin{cases} r_i, & i \in J \\ 0, & i \notin J \end{cases},\forall J \in \breve{\Im}(\mathbf{X})\right\}$$

The proposed algorithm shown in Schema 1, is based on the Lemmas of section 3 and the following Lemmas (they are proved in [8]).

SCHEMA 1
THE PROPOSED ALGORITHM

1. Enter the matrices $\mathbf{A}_{n\times m}$ and $\mathbf{b}_{n\times 1}$.
2. Compute the solution matrix Γ.
3. If there exists $j \in N_n$ such that $\Gamma_{\bullet j} = \mathbf{0}$ ($b_j \neq 0$), then $S(\mathbf{A},\mathbf{b}) = \varnothing$, since $S(\mathbf{A}_{j\bullet},b_j) = \varnothing$. Stop.
4. Compute the solution matrices $\hat{\Gamma}$ and $\breve{\Gamma}$.
5. If there exists $j \in N_n$ such that $\breve{\Gamma}_{\bullet j} = \mathbf{0}$ ($b_j \neq 0$), then $S(\mathbf{A},\mathbf{b}) = \varnothing$, since the j-th form (4) FRE is in contradiction with at least one of the others. Stop.
6. Compute $\hat{\mathbf{r}} = \inf\hat{\Gamma}$ (the maximum solution of the form (3) FRE).
7. Compute $\breve{\mathbf{r}} = \sup\breve{\Gamma}$ (the mean solution of the form (3) FRE).
8. Reduce the equation using Reduction 2.
9. (Reduce the equation using Reduction 1).
10. Compute $\Im_i(\breve{\Gamma}^*)$, $\Im(\breve{\Gamma}^*)$, $\breve{\Im}(\breve{\Gamma}^*)$, where $\breve{\Gamma}^*$ is the min-SM without the zero columns (and without the lines reduced by Reduction 1).
11. Compute $M(\breve{\mathbf{r}},\breve{\Gamma})$ (the minimal solution set $\breve{S}(\mathbf{A},\mathbf{b})$ of the equation).
12. (Extend the minimal solutions inserting zeros at the locations that were eliminated due to Reduction 1)
13. Stop.

Lemma 5. Let $\mathbf{A}\circ^t\mathbf{r} = \mathbf{b}$ be a FRE of the form (3) (reduced form), where t is a continuous t-norm. If $S(\mathbf{A},\mathbf{b}) \neq \varnothing$, then:

$$\forall \mathbf{r} \in M(\sup\breve{\Gamma},\breve{\Gamma}) \text{ it is } \mathbf{r} \in S(\mathbf{A},\mathbf{b})$$

where $\breve{\Gamma}$ is the min-SM of the FRE.

Lemma 6. Let $\mathbf{A}\circ^t\mathbf{r} = \mathbf{b}$ be a FRE of the form (3) (reduced form), where t is an Archemedean t-norm. If $S(\mathbf{A},\mathbf{b}) \neq \varnothing$, then $\mathrm{M}(\sup\breve{\Gamma},\breve{\Gamma})$, where $\breve{\boldsymbol{\Gamma}}$ is the min-SM of the FRE, is the set of the minimal solutions of the equation, i.e.

$$M(\sup\breve{\boldsymbol{\Gamma}},\breve{\boldsymbol{\Gamma}})=\breve{S}(\mathbf{A},\mathbf{b}).$$

A numerical example is now given, in order to make clear the mechanism of the method.

Let $\mathbf{A}\circ^{t}\mathbf{r}=\mathbf{b}$ be a FRE of the form (3), where t is the product operator, and

$$\mathbf{A}=\begin{bmatrix} .2 & .5 & .7 & 1 & .2 & 0\\ 0 & .1 & 1 & .4 & .3 & .2\\ 1 & .4 & .3 & 0 & 0 & .1\\ 1 & .6 & .2 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & .8 & 0 \end{bmatrix} \qquad \mathbf{b}=\begin{bmatrix} 1\\ .4\\ .7\\ .3\\ 0 \end{bmatrix}$$

The SM $\boldsymbol{\Gamma}$ is first calculated: $\boldsymbol{\Gamma}=\begin{bmatrix} 0 & 0 & .7 & .3 & 0\\ 0 & 0 & 0 & .5 & 0\\ 0 & .4 & 0 & 0 & 0\\ 1 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & .3 & 0 \end{bmatrix}$

Since $\boldsymbol{\Gamma}_{\bullet j}\neq\mathbf{0}$, $\forall j\in N_5$ with $b_j\neq 0$, we calculate the max-SM and min-SM:

$$\hat{\boldsymbol{\Gamma}}=\begin{bmatrix} 1 & 1 & .7 & .3 & 1\\ 1 & 1 & 1 & .5 & 1\\ 1 & .4 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 1\\ 1 & 1 & 1 & 1 & 0\\ 1 & 1 & 1 & .3 & 1 \end{bmatrix} \qquad \breve{\boldsymbol{\Gamma}}=\begin{bmatrix} 0 & 0 & 0 & .3 & 0\\ 0 & 0 & 0 & .5 & 0\\ 0 & .4 & 0 & 0 & 0\\ 1 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & .3 & 0 \end{bmatrix}$$

As it can be easily seen, the 3rd and the 5th columns of $\breve{\boldsymbol{\Gamma}}$ are zero-columns. Although for the 5th column, we have $b_5=0$, for the 3rd column, we have $b_3=.7\neq 0$, so the FRE of the form (3) has no solution, since the third FRE of the form (4) is in contradiction with at least one of the others.

Let now $\mathbf{A}=\begin{bmatrix} .2 & .5 & .7 & 1 & .2 & 0\\ 0 & .1 & 1 & .4 & .3 & .2\\ 1 & .4 & .3 & 0 & 0 & .1\\ 1 & .6 & .2 & 0 & 0 & 1\\ 0 & 0 & 0 & 0 & .8 & 0 \end{bmatrix} \qquad \mathbf{b}=\begin{bmatrix} .4\\ .2\\ .5\\ .7\\ 0 \end{bmatrix}$

We calculate the solution matrices $\boldsymbol{\Gamma}$, $\hat{\boldsymbol{\Gamma}}$ and $\breve{\boldsymbol{\Gamma}}$:

$$\Gamma=\begin{bmatrix}0 & 0 & .5 & .7 & 0\\ .8 & 0 & 0 & 0 & 0\\ .5714 & .2 & 0 & 0 & 0\\ .4 & .5 & 0 & 0 & 0\\ 0 & .6667 & 0 & 0 & 0\\ 0 & 1 & 0 & .7 & 0\end{bmatrix}\quad \hat{\Gamma}=\begin{bmatrix}1 & 1 & .5 & .7 & 1\\ .8 & 1 & 1 & 1 & 1\\ .5714 & .2 & 1 & 1 & 1\\ .4 & .5 & 1 & 1 & 1\\ 1 & .6667 & 1 & 1 & 0\\ 1 & 1 & 1 & .7 & 1\end{bmatrix}$$

$$\breve{\Gamma}=\begin{bmatrix}0 & 0 & .5 & 0 & 0\\ .8 & 0 & 0 & 0 & 0\\ 0 & .2 & 0 & 0 & 0\\ .4 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & .7 & 0\end{bmatrix}$$

The maximum and the mean solution are now found to be:

$$\hat{\mathbf{r}}=\begin{bmatrix}.5 & .8 & .2 & .4 & 0 & .7\end{bmatrix}^{\mathrm{T}} \text{ and } \bar{\mathbf{r}}=\begin{bmatrix}.5 & .8 & .2 & .4 & 0 & .7\end{bmatrix}^{\mathrm{T}}$$

We reduce the equation (we only ignore the 5th column of $\breve{\Gamma}$). Then, we determine the set $M(\bar{\mathbf{r}},\breve{\Gamma})$ (the set of the minimal solutions). We have

$$\Im_1(\breve{\Gamma})=\{2,4\},\ \Im_2(\breve{\Gamma})=\{3\},\ \Im_3(\breve{\Gamma})=\{1\} \text{ and } \Im_4(\breve{\Gamma})=\{6\}$$

$$\Im(\breve{\Gamma})=\{\{2,3,1,6\},\{4,3,1,6\}\}\quad \tilde{\Im}(\breve{\Gamma})=\{\{2,3,1,6\},\{4,3,1,6\}\}$$

and finally $\mathrm{M}(\bar{\mathbf{r}},\breve{\Gamma})=\left\{\begin{bmatrix}.5 & .8 & .2 & 0 & 0 & .7\end{bmatrix}^{\mathrm{T}},\begin{bmatrix}.5 & 0 & .2 & .4 & 0 & .7\end{bmatrix}^{\mathrm{T}}\right\}$

5 Conclusions

Speaking in fuzzy inference systems' (FIS's) terminology, equation (1) can represent a FIS $F(Y\to Z)$ based on the generalized modus ponens, where Y is the input space (discretized in m points), Z is the output space (discretized in k points) and X is the rule space (with cardinality n). In the present case **A** represents the determination of the fuzzy sets defined on Y (the left part of IF-THEN rules), **B** represents the determination of the fuzzy sets defined on Z (the right part of IF-THEN rules) and **R** represents a fuzzy implication $J(Y\to Z)$. The line vectors $\mathbf{A}_{j\bullet}$ and $\mathbf{B}_{j\bullet}$ ($\forall j\in N_n$), represent the fuzzy sets defined on Y and Z, respectively, and $\mathbf{A}_{j\bullet}\to\mathbf{B}_{j\bullet}$, $\forall j\in N_n$ represent the rules of the FIS. When equation (1) has a solution for **R**, the FIS satisfies the *modus ponens* rule of inference ($((A\to B)\wedge A)\to B$). This characteristic is very useful for the analysis of FIS. Lemma 1, Corollary 1, Lemma 2, and Corollary 2 are criteria for the solvability of (1), i.e. criteria for the satisfaction of the modus ponens. Furthermore, the proposed solution determination process (for any continuous t-norm) which is algorithmically simple, provides a way for implementing the FIS.

References

1. Gupta, M.M. and Qi, J., Design of fuzzy logic controllers based on generalized T-operators, Fuzzy Sets and Systems 40 (1991) 473-489.
2. Di Nola, A., et al., Fuzzy relation equations and their application to knowledge engineering (Kluwer Academic Press, Dordrecht, 1989).
3. Klir, G.J. and Yuan, B., Fuzzy sets and fuzzy logic: theory and applications (Prentice Hall PTR, USA, 1995).
4. Pappis, C.P., and Sugeno, M., Fuzzy relation equations and the inverse problem, Fuzzy Sets and Systems 15 (1985) 79-90.
5. Pedrycz, W., Processing in relational structures: fuzzy relational equations, Fuzzy Sets and Systems 25 (1991) 77-106.
6. Peeva, K., Fuzzy linear systems, Fuzzy Sets and Systems 49 (1992) 339-355.
7. Sanchez, E., Resolution of composite fuzzy relation equations, Information and Control 30 (1976) 38-48.
8. Stamou, G. and Tzafestas, S., Resolution of sup-t fuzzy relation equations, submitted.

An Approach to Generate Membership Function by Using Kohonen's SOFM Nets

Li Manping [1], Wang Yuying [1], and Zhang Xiurong [2]

[1] Southwest Electronic Engineering Institute, P.O.Box 517-11,
610003 Cheng Du, Si Chuan, P.R.China
leaiee@caepsc.ihep.ac.cn
[2] Hebei Institute of Civil Engineering and Architecture,
075024 Zhang Jia Kou City, He Bei, P.R.China

Abstract: This paper provides an approach that determines the fuzzy membership-degree based on Kohonen's Self-Organizing Feature Map (SOFM) nets and Lagrangian interpolation function. In this way, we need not depend on traditional membership function, but only use several typical models. It can be useful in distinguishing and processing some kinds of uncertain systems. The specific algorithm is given in paper, and at last, by means of data simulation, we get some objective and reasonable truth-values.

1 Introduction

Generation of membership function, is one of the basic problems of fuzzy theory. Along with application of fuzzy technology in all kinds of fields, many effective approaches on generating membership function have grown up. While compared with the probability distribution functions, they still show short of mature and reliability at some degree, which is an important reason that prevents fuzzy maths improving more quickly than it does now.

To make a specific membership-degree (MD), it's inevitable to contain man's work. So the result will always be somewhat subjective. Even if we abstract statistics information of experts, it's also hard to avoid prejudice. As to the method of complete statistics, the calculating job will be too complicate to apply. In practice, we want the membership function as objective as may be, and the calculation as easy as possible.

2 Feasibility

We find that, in a fuzzy set, there are always several typical models' MDs can win acceptance, but the others are uncertain. Target recognition in many conditions is like this. For example, there is one pattern target O, we are going to determine the degree of each unknown model $X^P(P=1, 2, \cdots)$in a universe of discourse U belongs to pattern target O. There are n features (factors) of model, i.e., $X^P = (X_1^P, X_2^P, \cdots, X_n^P)$, in all the unknown models X of U.

It's likely for us to pick up several typical models: X^I belongs to O exactly, X^J belongs to O roughly, and X^K does not belong to O at all. Say it in fuzzy theory language: These three typical models' MDs of O is 1, 0.6 and 0 respectively, these three MDs are accepted by us all. Accordingly, other undetermined MDs should be decided by these three ones.

Since there're many factors (n features) which affect the degree of a model X^Pbelonging to target O, it's difficult to give a membership function according to these three typical. In additional membership functions, the one-dimensional distribution function is certainly unfit for this multi-factors problem. The methods of statistics are also relatively hard to put. If all X^P in U are statistically calculated or contrasted one by one, that is to put every model (typical end others) in the same level. Then chances may be that the typical models' MDs are not equal to the accepted values, while the others cannot be test by a standard.

To make membership function of pattern recognizing problems like this, we hope find a method that satisfies the following three conditions:
(i). Simplify the problem into one-dimension.
(ii). Typical models' MDs are fit for the accepted ones.
(iii). Other models' MDs are generated according to the typical models'.

Kohonen's SOFM nets give the way to the job. In a SOFM net, after net's learning and training, topological structure of output space is similar to that of input space. If there are two models whose distance is close in input space, their location in output space is also close. So let the features that determine models' MDs be input values of SOFM net, which output neurons are queued in a line, that is a map from multi-dimension to one-dimension. Firstly, select several typical models as learning samples. After training, these few typical models are corresponded with few output points steadily. Now, weights of net are the number of those several input typical models, and their MDs are known by us. Then, input an undetermined model in universe of discourse U ; decide its MD on the basis of its location in output layer. If its location is close to certain typical model, its MD would be close to the known MD too. All the process of map is get from net's learning and training, so the result would be objective.

Up to now, problem has been simplified to one-dimension. To get the undetermined MDs, some interpolation functions or one-dimensional membership distribu-tion functions are feasible, such as Lagrangian Interpolation function to be used in this paper.

3 Specific Algorithm

Feature space is represented as n input neurons, the number of output neurons is n_1 $(n_1 >> n)$, they are queued in a line. Let $X^P = (X_1^P, X_2^P, \cdots, X_n^P)$ be input typical model vectors $(P = 1,\ 2,\ \cdots,\ K)$, there are K typical models in all. X_i^P is the ith feature associated with model X^P, and W_{ij} is the weight between ith feature (input neuron) and jth output neuron y_j. Function d show the match degree of vector X and weight $W_j = (W_{1j}, W_{2j}, \cdots, W_{nj})$.

$$d(X, W_j) = || X - W_j || \tag{1}$$

In the output layer, the neuron which match input vector X best is labeled with c, i.e.,

$$d(X, W_j) = \min_j d(X - W_j) \tag{2}$$

Modification of weight is applied to W_j (when $j \in N_c$), specific algorithm as following steps:

(1). Let initial values $W_j = W_j(0)$ are some random numbers.

(2). Select one of typical models $X^1, X^2, \cdots, X^K$ as input vector of Kohonen's SOFM net.

(3). Calculate

$$d = || X^P - W_j || = \sum_{i=1}^{n} (X_i^P - W_{ij}), \qquad j = 1,\ 2,\ \cdots,\ n_1 \tag{3}$$

Find $d_{\min}$ which correspond an output neuron y_c, y_c is called victory point.

(4). Modify related weights:

$$\begin{cases} W_j(t+1) = W_j(t) + a(t)(X - W_j(t)) & j \in N_c(t) \\ W_j(t+1) = W_j(t) & j \notin N_c(t) \end{cases} \tag{4}$$

Modifying field N_c can be larger at the beginning, about half of output layer, center point is y_c; then contract N_c progressively, modifying coefficient $a(t)$ is also reduced as time (t) goes on.

(5). Return to step (2), (3), (4), until every typical model is associated with a victory output neuron. Now, learning and training process has finished.

(6). Input an undetermined model X^U, find its victory point in output layer via calculating distance function d.

(7). Using Lagrangian Interpolation function, get the undetermined MD.

Let's describe step (7) in details. Lagrangian Interpolation function is:

$$f(x)=\sum_{i=1}^{n+1}\frac{b_i(x-a_1)\cdots(x-a_{i-1})(x-a_{i+1})\cdots(x-a_{n+1})}{(a_i-a_1)\cdots(a_i-a_{i-1})(a_i-a_{i+1})\cdots(a_i-a_{n+1})} \qquad \textbf{(5)}$$

$f(x)$ is the only polynomial whose degree is not larger than n, and satisfies the condition of $f(a_i)=b_i$(each $a_i$and $b_i$is known). So let victory points of$K$typical models be $a_1,\ a_2,\ \cdots,\ a_K$ respectively, their MDs are $b_1,\ b_2\ ,\cdots,\ b_K$, and the victory point of an undetermined model is X. Substitute them in this formula, then$f(x)$is what we want, undetermined MD. Now, MDs of typical models are equal to what we accept, while undetermined models' MDs are generated according to typical ones. This result has satisfied our hope.

4 Data Simulation

To test this method's effect, we give some data simulations. There are 4 input neurons, 40 output neurons which are corresponded with $0\sim 39$ in output line. The iteration formula of modifying field coefficient is:

$$\begin{cases} N_c(t)=21-t \\ a(t)=\dfrac{1}{2^t},\ \dfrac{1}{2^t+1},\ \cdots,\ \dfrac{1}{2^{t+1}} \end{cases} \qquad (t=1,\ 2,\ \cdots,\ 20) \qquad \textbf{(6)}$$

Select 4 typical models as input: $X^1,\ X^2\ ,X^3\ ,X^4$, their accepted MDs are $T^1,\ T^2\ ,T^3\ ,T^4$ respectively. After net's training, the related victory output points are $Y^1,\ Y^2\ ,Y^3\ ,Y^4$. X_i^U, Y_i^U and T_i^U are represented the ith undetermined model's input vector , its victory output point and its MD respectively.

(1). $X^1, X^2\ ,X^3\ ,X^4$: $(0.1\quad 0\quad 0.08\quad 0)$,
$(4.04\quad 3.96\quad 4.1\quad 3.8)$,
$(7.03\quad 6.95\quad 7.1\quad 6.89)$,
$(10.1\quad 9.97\quad 10.06\quad 9.8)$

$T^1,\ T^2\ ,T^3\ ,T^4$: 0, 0.4, 0.7, 1

$Y^1,\ Y^2\ ,Y^3\ ,Y^4$: 0, 11, 23, 36

a. $X_a^U,\ Y_a^U,\ T_a^U$: $(2.1\quad 3.13\quad 4.01\quad 1.3)$, 5, 0.27

b. $X_b^U,\ Y_b^U,\ T_b^U$: $(8.07\quad 7.94\quad 8.2\quad 7.83)$, 28, 0.811

(2). X^1, X^2 ,X^3 ,X^4: (0.99 9.7 103 1008),
(4.87 49.4 510 5009),
(8.01 78.8 806 8016),
(10.05 96.8 1012 10024)

T^1, T^2 ,T^3 ,T^4: 0.1, 0.5, 0.8, 1

Y^1, Y^2 ,Y^3 ,Y^4: 1, 12, 25, 39

a. X_a^U, Y_a^U, T_a^U: (2.5 30 310 2611), 6, 0.28

b. X_b^U, Y_b^U, T_b^U: (8 60 810 8016), 18, 0.68

5 Conclusions

In this paper, we provide an approach that determines the fuzzy membership-degree based on Kohonen's Self-Organizing Feature Map (SOFM) nets and Lagrangian interpolation function. The specific algorithm and part result of data simulation are given. It can be seen that the truth-values are objective and reasonable.

From these data, we find that when each feature of undetermined model is not at the same level (such as (2).b.), This approach shows its advantage in recognizing and classifying, because on this condition, the distance between X^U and typical model is difficult to decide, while using Kohonen's SOFM net, the result will be objective. On the other hand, this method also has shortcoming what other SOFM nets have. That is, when map is done from high-dimension to low-dimension, if the deflation ratio is too large, there would be distortion. To treat this situation, multi-map may be available. More complex problems would be included in future work.

References

1. L.A.Zadeh, Fuzzy sets, Inf. Control 8, (1965) 338-353
2. Zhang Liming, Models and Application of Artificial Neural Networks, Fudan University Press, (1993)
3. T.Kohonen, Adaptive Associative and Self-Organizing Function in Neural Computing Applied Optics Vol. 26, (1987) 4910-4918
4. He Zhongxiong, Fuzzy Maths and Its Applications, Tianjin Science and Technology Press, (1983)
5. Kwan.C.M, Robust spacecraft attitude control using fuzzy CMAC, Proceedings of the 1996 IEEE International Symposium on intelligent Control, (1996) 43-50

Intelligent Policing Function for ATM Networks

Anna Rita Calisti, Francisco David Trujillo Aguilera, Antonio Díaz Estrella and
Francisco Sandoval Hernández

Departamento Tecnología Electrónica, E.T.S.I. Telecomunicación,
Campus de Teatinos, 29071 Málaga, Spain.
trujillo@dte.uma.es

Abstract. In this work we present an intelligent system implemented with a hybrid technique that aims to resolve the problem of traffic control in Asynchronous Transfer Mode (ATM) networks. In this system, fuzzy and neural techniques are combined to obtain a policing mechanism simple enough to be cost-effective and with a performance hugely better than the traditional ones. The UPC proposed is very fast, has a high selectivity and presents an immediate hardware implementation.

1 Introduction

Artificial intelligence techniques are more and more explored in the Communications field [1]. In this work we present an intelligent system implemented with a hybrid technique that aims to resolve the problem of traffic control in Asynchronous Transfer Mode (ATM) networks.

ATM is the transport mode chosen for Broadband Integrated Service Digital Network (B-ISDN). In the ATM networks, the information is packed in 53 bytes fixed length cells. One of ATM network designer challenge is to develop a framework that maximizes the utilization of network resources and, at the same time, controls the traffic flow in the network so as to avoid periods of congestion, which will cause the network performance to degrade dramatically. For this purpose, several preventive controls are used, namely, the Connection Admission Control (CAC) and the Usage Parameter Control (UPC) [2]. CAC decides, during the call set-up phase, to accept a new connection if it considers (relying on the negotiated Traffic Parameters [2] specified in the Traffic Contract) that the required Quality of Service (QoS) can be guaranteed for both connections already established and for the new one. The Traffic Parameters, such as the peak or the mean speed, describe an inherent characteristic of the traffic source. The UPC function, instead, controls the cell stream during the entire active phase of the call and restricts the behaviour of the traffic source into the network to the characteristics negotiated in the contract.

The purpose of the UPC is to protect network resources from malicious as well as unintentional misbehaviour which can affect the QoS of the already established connections by detecting violations of negotiated parameters and taking appropriate action, such as cells-marking or cells-dropping. An UPC algorithm should have the following features: 1) high selectivity, that is the capability to detect any non-compliant traffic situation, and transparency for connections, that respect the parameter values negotiated, 2) rapid response to parameter violations and 3) simplicity of implementation and cost-effectiveness.

To meet these somewhat controversial requirements, several policing mechanisms have been proposed [3], such as the Leaky Bucket (LB) recommended by ITU-TS [2] (which consists of a counter of size L increased on the arrival of the cells and decreased, if positive, at a constant frequency λ_p), window mechanisms (which limit the number of cells accepted from a source, within a fixed time interval T to a maximum number N), and several others. Figure 1 shows the simplified scheme of a LB and a Jumping Window (JW).

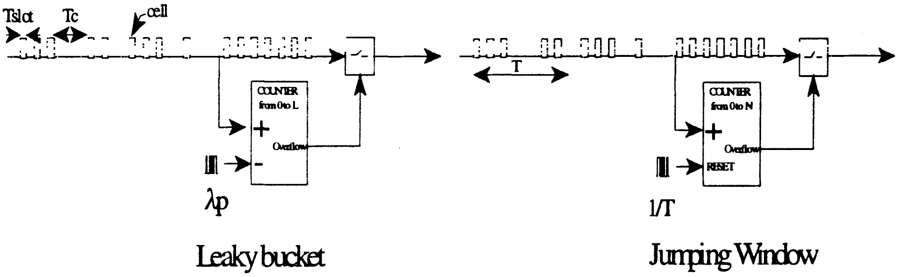


Fig. 1. Leaky bucket and Jumping Windows mechanisms

Peak Cell Rate (PCR) and Sustainable (mean) Cell Rate (SCR) are the Traffic Parameters usually policed. Policing of the PCR is not generally complex and can be easily achieved by using a LB. However, enforcement of the SCR is an open issue [4-6]. The UPC algorithm should enforce the mean cell rate by allowing short-term statistical fluctuations of the source traffic as long as the source respects the average value negotiated in the long term. None of the traditional mechanisms seem to be capable of policing the SCR fulfilling the above-mentioned controversial requirements.

Selectivity can be improved, for instance, either increasing the observation period or increasing the mechanism implementation complexity [4]. However, this has been shown to be in conflict with the responsiveness and simplicity requirements. The limit of the traditional mechanisms lies on the a priori static choice of control parameters which defined them (e.g., LB parameters are the counter size L and the depletion rate λ_p). Fixed values of the thresholds are not suitable to control stochastic magnitudes, that are allowed to fluctuate around an average value. For example, the LB threshold L should be high enough to reduce the false alarm probability, that is, the probability of detecting some cells of a non-violating source as excessive and, at the same time, short enough to reduce the reaction time.

The basic idea introduced in this work is to use a Neural Net (NN) to update such control parameters during the entire phase of the control depending on the behaviour of the traffic source. The introduction of a NN leads to an UPC mechanism very near to the ideal one at a cost of a very small hardware complication. Some papers [5][6] that propose applying a fuzzy approach to traffic control in ATM networks have already been published. This work proposes an UPC mechanism based on a very simple NN instead of a fuzzy controller, obtaining a high improvement in terms of hardware simplification (the NN has just three elaborating neurons) and processing speed. Another advantage is that, besides empirical rules, QoS and traffic measures in ATM networks can be used as inputs to the NN in order to achieve a better control [7]. This paper contains four Sections. In Section 2 the new model of UPC and a discussion of the empirical rules used as inputs to the NN are shown. Section 3 is divided into three parts. In the first one all the control parameters are defined; in the second one the learning of the NN is explained and, in the third part, the performance evaluation and a comparison with the LB are presented. Finally, some conclusions and future lines are drawn in Section 4.

2 Intelligent UPC proposed

In our approach, control is based on global evaluation of the source behaviour from the beginning of the connection up to the instant when the control is exercised. The intelligent UPC (IUPC) proposed (figure 2) is a JW-based control mechanism in which the maximum number N_i of cells that can be accepted in the *ith* window of length T is a threshold dynamically update by a NN, previously trained.

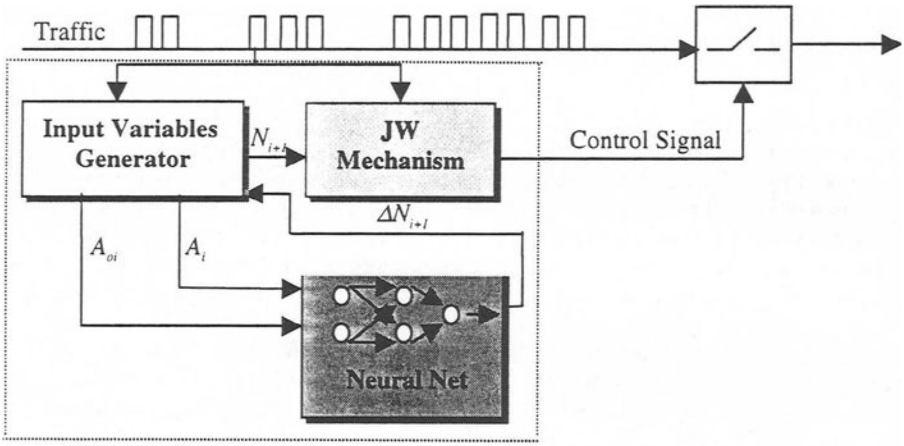


Fig. 2. Intelligent UPC

The NN updates the threshold N_i, according to the value of two parameters, A_{oi} and A_i, which give an indication of the long-term trend of the source and its current behaviour, respectively. More exactly, A_{oi} represents the average number of cell arrivals per window from the start of the connection and A_i the number of cell arrivals in the last window. The output ΔN_{i+1} represents the variation to be made to the threshold N_i, called credit, in the next window: $N_{i+1}=N_i+\Delta N_{i+1}$.

In the training phase the NN learns from empirical rules, which can be established depending on A_{oi} and A_i values.

If A_{oi} is much smaller than the expected value of cells per windows, $N=T\times SCR$, that means a source fully respectful. In this case, if Ai is near to N the source presents non-violating behaviour and its credit is increased. But if A_i is much greater than N (a sign of a possible beginning of violation or an admissible short-term statistical fluctuation), the threshold value remains unchanged.

If A_{oi} is near to N, that means a source moderately respectful. If $A_i \leq N$, the control mechanism grants credit to that source that, at the present, respects the parameter negotiated; otherwise, the credit is reduced.

When A_{oi} is much greater than N, the source is violating. If A_i is smaller than N, the threshold must be immediately raised to avoid excessively rigorous policing. If A_i is near to N, the threshold remains constant. Finally, if A_i is greater than N, the threshold must decrease because the source continues violating the contract.

According to the empirical rules, we propose in figure 3 a training pattern set to be presented to the NN in the learning process.

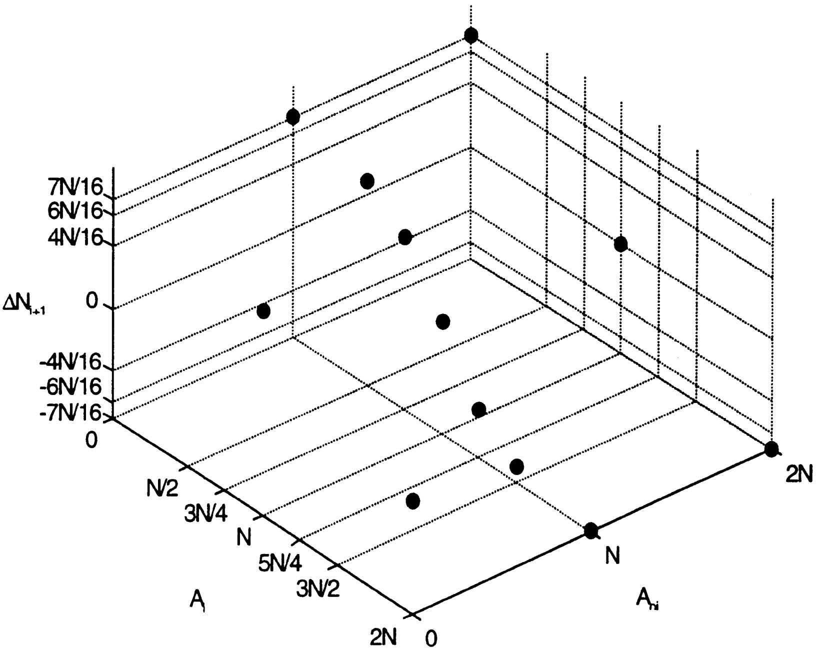


Fig. 3. Training pattern set. Each dot is a training pattern

The training patterns could be mapped with a non-linear continuous surface which can be easily obtained with a Feedforward Multilayer Perceptron with back-propagation learning algorithm [8].

3 Simulation and results

In this section we compare the performance of our IUPC and the LB mechanism, being this the chosen one by ITU-TS [2] to implement the policing function. For this purpose, we consider a bursty source with an exponential distribution for the duration of both periods, with a mean of T_{on}=174 μs for the active phase and T_{off}=243.42 μs for the idle phase. The intercell time during a burst is T_c=34.6 μs.

3.1 Control Parameters

The parameters of our IUPC are *T*, *N* and the upper bound and initial values for the credit, N_i.

The chosen length for the window is *T*=2671.48 μs which corresponds near to 6×(T_{on} + T_{off}). The expected value of cells per window is $N=T\times SCR$ = 32 cells, where SCR is $T_{on}/((T_{on}+T_{off})\times T_c)$. The selected value for the upper bound of N_i is one of the main issues in sizing the mechanism. It has to take into account two conflicting requirements. The first one requires a high value for N_i to reduce the false alarm probability. The second one requires a low value for N_i to improve rection time. The best trade-off between responsiveness and false alarm probability was obtained choosing 19*N* as the maximum value of N_i. The same reasoning leads to fix, at the beginning of connection, N_i= 4*N* as its first value. Finally, the best parameters of the LB for the source mentioned above are: counter size *L*=300 cells and depletion rate $\lambda_p=C\times SCR$,with *C* = 1.1.

3.2 Learning

The learning-rate parameter and the momentum constant values for the NN are 0.15 and 0.075, respectively. The normalization process depends on the maximum number of cells that can arrive in a window (T/T_c=77≈80 cells) and the maximum value of ΔN_{i+1} (± 7*N*/16 = ±14) (see figure 3). This leads us to choose the values shown in table 1 as the normalized ones.

Table 1. Parameter normalization

A_{oi}	a_i	Dn_{i+1}
$A_{oi}/80$	$A_i/80$	$(\Delta N_{i+1}+14)/40+0.1$

After 40000 iterations, the error takes the sufficiently small value of 2x10^{-6}. Figure 4 shows the NN surface learnt, where we can see that the NN has reconstructed a surface which covers all the possible values of A_{oi} and A_i.

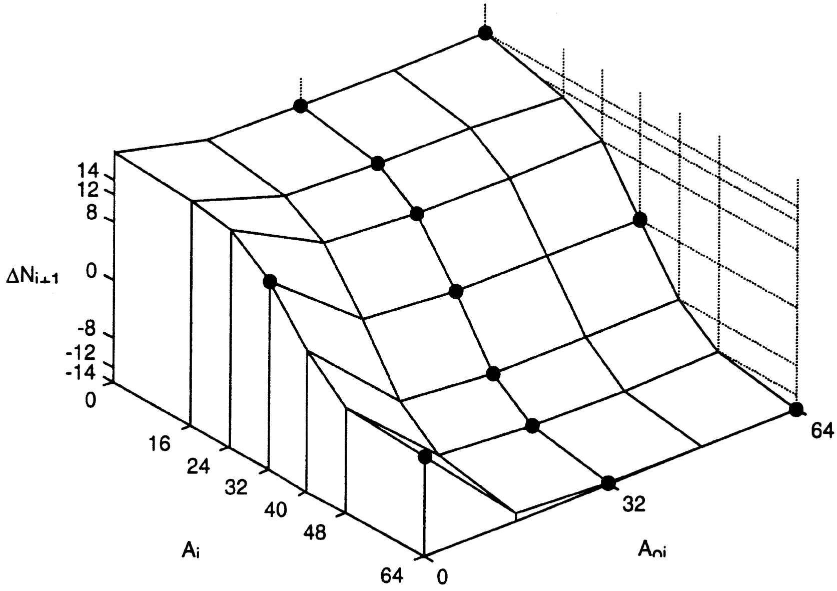


Fig. 4. Surface learnt by the NN

3.3 Performance evaluation

We define P_d as the probability that the policing mechanism detects a cell as excessive. P_d should be zero if the mean cell rate is less or equal to the nominal one, and $P_d=(Y-1)/Y$ for $Y>1$, where Y is the long-term actual mean cell rate of the source normalized to the SCR. In order to obtain P_d versus Y, we assume that a variation in the cell rate is due to a change in the average number of cells for burst, while the average silence time is assumed to be constant.

As it is shown in figure 5, the IUPC presents a null false alarm probability for $Y=1$ and, in the case of violating sources ($Y>1$), a probability of detection almost coincident with the ideal one.

In figures 6 and 7 the dynamic behaviour of the two policing mechanisms has been compared. Figures 6 and 7 show the responsiveness in terms of the fraction of violating cells detected versus the average number of cells emitted by a source with an actual mean cell rate normalized $Y=1.09$, and $Y=1.64$, respectively.

In figure 6 it is noticed that IUPC begins to detect violations after 600 cells, while LB does it after 1200. Moreover, the IUPC detection probability grows very fast and only after 2400 cells reaches a value of about 5.23% against the ideal 8.2%, showing a marked improvement over the LB mechanism.

In figure 7, this trend is confirmed for Y=1.64: after 2400 cells IUPC percentage of cells detected as excessive is in the range of 32%, against the ideal 39% and the 27% for LB.

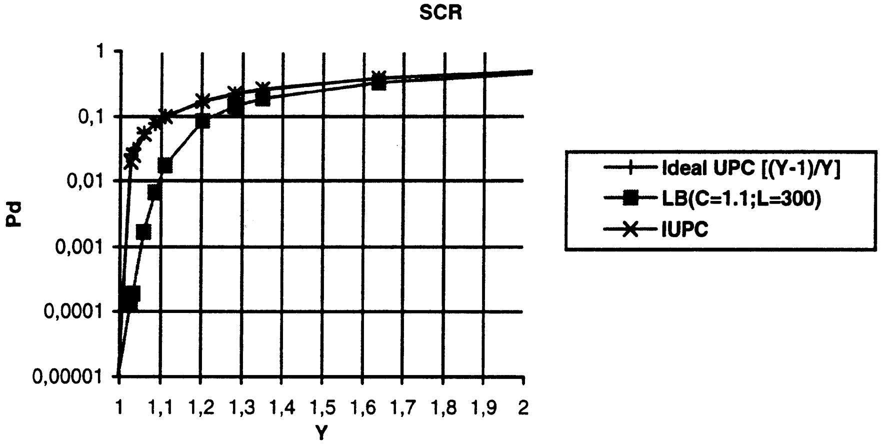


Fig. 5. Selectivity performance versus cell rate variations

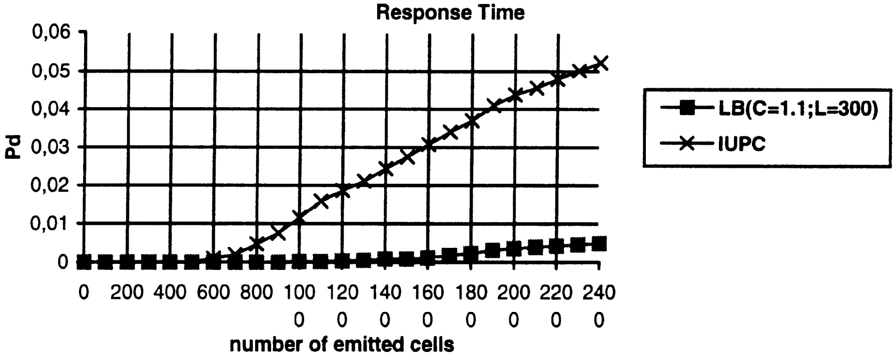


Fig. 6. Dynamic behavior for Y = 1.09

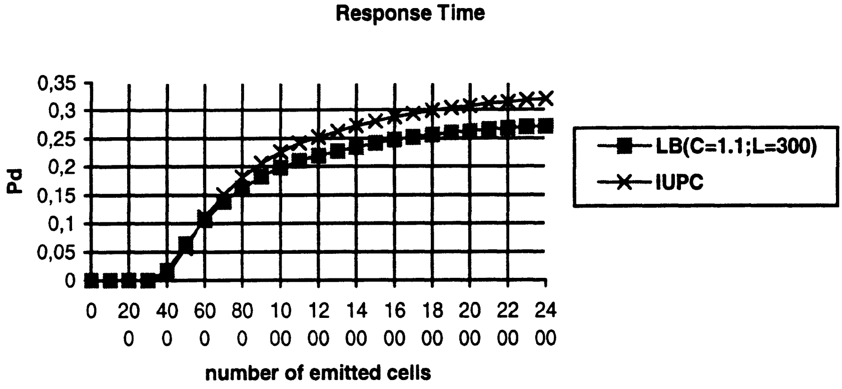


Fig. 7. Dynamic behavior for Y = 1.64

4 Conclusions

In this paper, we have introduced a UPC based on NN for ATM networks that combines fuzzy and neural techniques. As we have explained before, this UPC is very fast, has a high selectivity and presents an immediate hardware implementation. Besides, it can be easily enlarged to improve the traffic control. In this sense, additional inputs could give information about the state or the trend of the output buffer occupancy in an ATM node, in order to achieve a better control taking into account the statistical multiplexion of the traffic source in the node.

Acknowledgments

This work has been partially supported by the *Spanish Comisión Interministerial de Ciencia y Tecnología (CICYT)*, project n. TIC96-0743.

References

1. PARK, Y. and LEE G., "Applications of Neural Networks in High-Speed Communication Networks", *IEEE Communications Magazine*, pp. 68-74, October 1995..
2. ITU-T, *Recommendation I.371-Traffic control and congestion control in B-ISDN*, Geneve, 1996.
3. RATHGEB, E., "Modelling and performance comparison of policing mechanism for ATM networks", *IEEE Journal on Selected Areas in Communications*, vol. 9, n. 3, pp. 325-334, 1991.
4. HLUCHYJ M.G., and YIN, N., "A second-order Leaky Bucket Algorithm to Guarantee QoS in ATM Networks", *Proceedings of GLOBECOM'96*, vol. 2, pp. 1090-1096, London, 1996.
5. NDOUSSE, T. D., "Fuzzy neural control of voice cells in ATM networks", *IEEE Journal on Selected Areas in Communications*, vol. 12, n. 9, pp. 1488-1494, 1994.
6. CATANIA, V., FICILI, G., PALAZZO, S. and PANNO, D., "A comparative analysis of fuzzy versus conventional poling mechanism for ATM networks", *IEEE/ACM Transactions on Networking*, vol. 4, n. 3, pp. 449-459, 1996.
7. GARCÍA, C., TRUJILLO, F. D., CALISTI, A., DÍAZ, A., and SANDOVAL, F., "Funciones UPC para tráfico VBR basadas en técnicas de inteligencia artificial", *Proceedings of JITEL'97*, pp. 119-127, Bilbao, September 1997.
8. RUMELHART, D., HINTON, G. y WILLIAMS, R., "Learning internal representations by error propagation", in *Parallel distributed processing: exploration in the microstructure of cognition, (Rumelhart, D.E. and McClelland, J.L., Eds.)*, MIT Press, pp. 318-362, 1986.

Deriving Fuzzy Subsethood Measures from Violations of the Implication between Elements

Francisco Botana

Departamento de Matemática Aplicada, Universidad de Vigo, Campus A Xunqueira,
36005 Pontevedra, Spain

Abstract. The aim of this paper is to present a collection of new measures of subsethood between fuzzy sets. Starting from the relationship between crisp set containment and logical implication, some fuzzy approaches are reviewed. An excerpt of reasonable fuzzy implication operators is used to define fuzzy measures of inclusion using Kosko's *fit-violation strategy.* We test these measures on two axiomatics and derive, when possible, measures of fuzzy entropy. Once a subsethood measure between fuzzy sets is defined, other operations as set equality, similarity, disjointness, complement, ... can be considered. The need for containment measures is present in wide areas as approximate reasoning and inference, image processing or learning.

1 Introduction

In any theory of sets, being either crisp or fuzzy, the inclusion operator is found among the first places of the basic definitions. The axiomatization usually chosen in crisp set theory, that is, Zermelo-Fraenkel's set theory [6], states the equality between sets as its first axiom, the *axiom of extension*:

$$(\forall a)(\forall b)\left((\forall x)(x \in a \leftrightarrow x \in b)\right) \rightarrow a = b \ . \tag{1}$$

Replacing $\leftrightarrow$ with $\rightarrow$ in (1) is the usual way to define set inclusion

$$a \subset b \equiv (\forall x)\,(x \in a \rightarrow x \in b) \ . \tag{2}$$

On his seminal paper of 1965 Zadeh [18] defines a fuzzy set in a universe, the equality among fuzzy sets and third the inclusion operator: given A, B fuzzy sets in a universe X, B *contains* A, $A \subset B$, if, and only if, $\mu_A(x) \leq \mu_B(x)$, $\forall x \in X$.

The consensus about the inclusion operator in crisp set theory does not remain in the fuzzy framework. Zadeh's fuzzy containment is a crisp property. Bandler and Kohout point it out in this way; they think that notion "represents, however, an unconscious step backward to the realm of dichotomy. [...] far more in accord with the spirit of fuzzy theory would be the assignment of a *degree* to A's 'subsetness' in B" [1, p. 13]. Dubois and Prade state that "Zadeh's definition of inclusion may appear very strict" [3, p. 22]. For Kosko "either [a] fuzzy set A is, or is not, a fuzzy subset of B. So Zadeh's relation of fuzzy subsethood is *not* fuzzy. It is black or white" [8, p. 279]. Because of that a great number of fuzzy alternatives to Zadeh's original operator have been suggested in literature.

In Section II we recall what is meant with fuzzy subsethood measure, we list some of them and summarize the axiomatic systems proposed by Sinha and Dougherty [12] and Young [17]. Section III describes in which way measures of subsethood have been derived from fuzzy implication operators and presents a set of new measures, studying which axioms they fulfill. The obtaining of these measures comes from Kosko's observation [8] about the relationship between his well-known measure and the implication operator of Łukasiewicz.

2 Fuzzy Subsethood Measures

Throughout this paper X is the universe on which fuzzy sets are defined. We assume that X is a finite set, but one can readily generalize the results for X infinite. We adopt the maximum and minimum operations for union and intersection.

The measure of containment between two fuzzy sets A, B in X proposed by Zadeh is a crisp binary relation in the set of fuzzy sets in X, $\mathcal{F}(X)$,

$$\begin{array}{l} \subset : \mathcal{F}(X) \times \mathcal{F}(X) \to \{0,1\} \\ \quad (A\ ,\ B) \quad \to A \subset B = \begin{cases} 1 & \text{if } \mu_A(x) \le \mu_B(x) \quad \forall x \in X \\ 0 & \text{in other case} \end{cases} \end{array} .$$

This crispness has led to a number of alternatives. An option consists of relaxing the containment to a certain degree, yet maintaining its crisp character. In [3] the *weak inclusion* is defined among fuzzy sets: A is *α–weakly included in B*, $A - \prec_\alpha B$, if all the elements of X belong to the α–cut of $\bar{A} \cup B$. Note that if we want the weak inclusion to generalize the inclusion of Zadeh α must be $\le \frac{1}{2}$.

The way usually chosen deals with the fuzzification of Zadeh's definition. So, a generic measure of subsethood, also called *inclusion grade* or *indicator for fuzzified set inclusion*, is a fuzzy binary relation in $\mathcal{F}(X)$, i.e., a fuzzy subset in $\mathcal{F}(X) \times \mathcal{F}(X)$

$$S : \mathcal{F}(X) \times \mathcal{F}(X) \to [0,1] \ ,$$

such that it extends the classic definition, $S(A,B) = 1$ if $A \subset B$, where $A, B \in \mathcal{F}(X)$.

Many subsethood measures have been proposed in literature. Sánchez [11] proposes one based on cardinality and intersection

$$S(A,B) = \frac{|A \cap B|}{|A|} \ . \tag{3}$$

Based on inclusion and cardinality Goguen [5] gives the measure

$$S(A,B) = |\overline{(A| - |B)}| \ , \tag{4}$$

where $| - |$ is the ***bounded difference*** defined by $\mu_{A|-|B}(x) = \max(0, \mu_A(x) - \mu_B(x))$. A measure derived from (4), based only on inclusion, is

$$S(A,B) = \inf_{x \in X} \mu_{\overline{A|-|B}}(x) \ . \tag{5}$$

Kosko [8] sets out again the measure (3) originally proposed by Sánchez, getting it twofold, in a geometric and algebraic way. Furthermore, he defines the fuzziness of a fuzzy subset A as the subsethood of $A \cup \bar{A}$ in $A \cap \bar{A}$, which satisfies the axioms of fuzzy entropy of De Luca and Termini [2].

Two axiomatizations for fuzzy subsethood measures have been proposed. In [12] nine basic properties are stated as axioms for an indicator for fuzzified set inclusion. The SD axioms are

1. $S(A,B) = 1$ if, and only if, $A \subset B$.
2. $S(A,B) = 0$ if, and only if, exists $x \in X$ such that $\mu_A(x) = 1$ and $\mu_B(x) = 0$.
3. If $B_1 \subset B_2$ then $S(A,B_1) \leq S(A,B_2)$.
4. If $A_1 \subset A_2$ then $S(A_2,B) \leq S(A_1,B)$.
5. $S(A,B) = S(\mathcal{L}(A),\mathcal{L}(B))$ where $\mu_{\mathcal{L}(A)}(x) = \mu_A(s(x)) \quad \forall x \in X$ being $s : X \to X$ any one–to–one mapping.
6. $S(A,B) = S(\bar{A},\bar{B})$.
7. $S(A_1 \cup A_2,B) = \min(S(A_1,B),S(A_2,B))$.
8. $S(A,B_1 \cap B_2) = \min(S(A,B_1),S(A,B_2))$.
9. $S(A,B_1 \cup B_2) \geq \max(S(A,B_1),S(A,B_2))$.

From these properties the authors postulate an expression as indicator and derive the conditions under which the expression satisfies the axioms. A class of indicators is obtained

$$S(A,B) = \inf_{x \in X}\min[1,\lambda(\mu_A(x)) + \lambda(1-\mu_B(x))] \ , \qquad (6)$$

where examples of the λ function are $\lambda(x) = 1-x^n$ for $n \geq 1$ real or $\lambda(x) = \frac{1-x}{1+nx}$ for $-1 < n \leq 0$ real.

Young [17] sets out her axiomatization (Y) imposing that the measure $E(A) = S(A \cup \bar{A}, A \cap \bar{A})$ will be a fuzzy entropy. Her axioms are

1. $S(A,B) = 1$ if, and only if, $A \subset B$.
2. If $P \subset A$ then $S(A,\bar{A}) = 0$ if, and only if, $A = X$, where $\mu_P(x) = \frac{1}{2} \quad \forall x \in X$.
3. If $B \subset A_1 \subset A_2$ then $S(A_1,B) \geq S(A_2,B)$.
4. If $B_1 \subset B_2$ then $S(A,B_1) \leq S(A,B_2)$.

3 Fuzzy Implication Operators and Measures of Subsethood

Recall that (2) is the usual way to define crisp set inclusion. So, inclusion relates to logical implication. In the fuzzy case, we can define the containment between A and B measuring in some way the implication degree $A \to B$. This is the case in [17]: fuzzy set inclusion is defined through

$$S(A,B) = \frac{1}{|X|}|A \to B| = \frac{1}{|X|}\sum_{x \in X}(1 - \mu_A(x) + \mu_A(x)\mu_B(x)) \ , \qquad (7)$$

where $\mu_{A \rightarrow B}(x) = (1 - \mu_A(x) + \mu_A(x)\mu_B(x))$. This approach is similar to the one taken in [1]: in the crisp case A is a subset of B if, and only if, $A \in 2^B$, being 2^B the power set of B. The measure of containment of a fuzzy subset A in B is the degree of membership of A in the fuzzy power set of B, $\mathcal{F}(2^B)$. Therefore, given a fuzzy implication operator $\rightarrow$,

$$S(A,B) = \mu_{\mathcal{F}(2^B)}(A) = \inf_{x \in X}(\mu_A(x) \rightarrow \mu_B(x)) \ . \tag{8}$$

The harsh criterion conveyed by the infimum is softened in [14], where the containment is taken as the mean of the implication between elements, in a similar way to the one taken in (7).

The algebraic derivation of Sánchez–Kosko's measure (3) is based on measuring in which degree the inclusion $A \subset B$ for each element $x \in X$ is not fulfilled. $\max(0, \mu_A(x) - \mu_B(x))$ is used to establish the violation of the implication for each element. In [8, p. 281] the relationship between its measure and Łukasiewicz's implication operator is emphasized:

$$1 - \max(0, \mu_A(x) - \mu_B(x)) = \min(1, 1 - \mu_A(x) + \mu_B(x)) \ . \tag{9}$$

A similar approach can be taken systematically with other fuzzy implication operators in order to obtain new measures of containment. A lot of operators have been considered in literature. Since we want the subsethood measure to be reduced to the ordinary inclusion in the case of crisp values, we have rejected those operators which do not fulfill the fourth axiom of [13], being this axiom as follows

$$(\forall a, b \in [0,1])\,((a \leq b) \leftrightarrow (a \rightarrow b = 1)) \ . \tag{10}$$

In case of equality, $a = b$, we want $a \rightarrow a$ to be a *strong tautology*, meaning that its value is 1, no matter which a's value is, as pointed in [1]. So, well-known operators as Kleene–Dienes ($a \rightarrow b = \max(1 - a, b)$) [7], Mamdani ($a \rightarrow b = \min(a, b)$) [9] or Yager ($a \rightarrow b = b^a$) [16] will not be considered.

We have chosen five operators, shown in Table 1. With each we remember their definition, we show their behaviour in the decile set $V_{11} = \{0, 0.1, \ldots, 0.9, 1\}$, we define a measure of subsethood and study which axiomatizations they fulfill.

3.1 The Wu Operator

This operator is presented in [15]. Note the 1's in the main diagonal and upper in Table 2, meaning that (10) is satisfied. To measure the containment of A in B we proceed as follows. For each $x \in X$ no restriction for containment exists if $\mu_A(x) \leq \mu_B(x)$. If $\mu_A(x) > \mu_B(x)$ the number $\overline{\mu_A(x) \rightarrow_{\mathrm{W}} \mu_B(x)} = 1 - \min(1 - \mu_A(x), \mu_B(x)) = \max(\mu_A(x), 1 - \mu_B(x))$ represents the violation of $A \subset B$ with regard to $x \in X$. Summing and dividing by the cardinality of X, we get a normalized measure of inclusion violations. Therefore,

$$S_{\mathrm{W}}(A,B) = 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_A(x) > \mu_B(x)}} \max(\mu_A(x), 1 - \mu_B(x)) \ , \tag{11}$$

Table 1. Fuzzy implication operators used to derive subsethood measures

No	Name	Definition
1	Wu W	$a \rightarrow_{\mathrm{W}} b = \begin{cases} 1 & \text{if } a \leq b, \\ \min(1-a, b) & \text{in other case} \end{cases}$
2	Goguen G43	$a \rightarrow_{\mathrm{G43}} b = \begin{cases} 1 & \text{if } a \leq b, \\ \frac{b}{a} & \text{in other case} \end{cases}$
3	Modified Goguen mG43	$a \rightarrow_{\mathrm{mG43}} b = \begin{cases} \min(1, \frac{b}{a}, \frac{1-a}{1-b}) & \text{if } a > 0 \text{ and } b < 1, \\ 1 & \text{in other case} \end{cases}$
4	Gödel S*	$a \rightarrow_{\mathrm{S^*}} b = \begin{cases} 1 & \text{if } a \leq b, \\ b & \text{in other case} \end{cases}$
5	Standard Strict S	$a \rightarrow_{\mathrm{S}} b = \begin{cases} 1 & \text{if } a \leq b, \\ 0 & \text{in other case} \end{cases}$

Table 2. W operator for value set V_{11}

W	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1	1	1	1	1	1	1	1	1	1	1
0.1	0	1	1	1	1	1	1	1	1	1	1
0.2	0	0.1	1	1	1	1	1	1	1	1	1
0.3	0	0.1	0.2	1	1	1	1	1	1	1	1
0.4	0	0.1	0.2	0.3	1	1	1	1	1	1	1
0.5	0	0.1	0.2	0.3	0.4	1	1	1	1	1	1
0.6	0	0.1	0.2	0.3	0.4	0.4	1	1	1	1	1
0.7	0	0.1	0.2	0.3	0.3	0.3	0.3	1	1	1	1
0.8	0	0.1	0.2	0.2	0.2	0.2	0.2	0.2	1	1	1
0.9	0	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	1	1
1	0	0	0	0	0	0	0	0	0	0	1

can be used to measure the subsethood between A and B.

It is easy to prove the following

Proposition 1. *S_{W} satisfies the Y axioms and does not satisfy the SD axioms.*

Neither S_{W} nor the measures of the following subsections satisfy the SD axioms. All of them fail at least on (SD2). Sinha and Dougherty are conscious of the unnaturality of this axiom but argue that it is consistent with the nuances of fuzzy set theory. In spite of this there are failures in other axioms. For example, S_{W} does not satisfy (SD6) either.

Due to Prop. 1 S_{W} is a fuzzy measure of subsethood in the sense of Young and

$$\begin{aligned} E_{\mathrm{W}}(A) &= S_{\mathrm{W}}(A \cup \bar{A}, A \cap \bar{A}) \\ &= 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_{A\cup\bar{A}}(x) > \mu_{A\cap\bar{A}}(x)}} \max(\mu_{A\cup\bar{A}}(x), 1 - \mu_{A\cap\bar{A}}(x)) \end{aligned} \tag{12}$$

$$= 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_A(x) \neq 0.5}} \max(\mu_A(x), 1 - \mu_A(x))$$

is the entropy of fuzzy set A in the sense of DeLuca and Termini.

3.2 The Goguen Operator

Recommended in [4] for further study, Table 3 shows its values in $\mathbb{Q}$ with error less than 0.1 when evaluated in V_{11}^2. For $x \in X$ and $\mu_A(x) \neq 0$, $\max(0, 1 - \frac{\mu_B(x)}{\mu_A(x)})$ is the violation of inclusion. So the measure obtained is

$$S_{\mathrm{G43}}(A, B) = 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_A(x) \neq 0}} \max(0, 1 - \frac{\mu_B(x)}{\mu_A(x)}) \ . \tag{13}$$

Table 3. G43 operator for value set V_{11}

G43	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1	1	1	1	1	1	1	1	1	1	1
0.1	0	1	1	1	1	1	1	1	1	1	1
0.2	0	0.5	1	1	1	1	1	1	1	1	1
0.3	0	0.3	0.7	1	1	1	1	1	1	1	1
0.4	0	0.3	0.5	0.8	1	1	1	1	1	1	1
0.5	0	0.2	0.4	0.6	0.8	1	1	1	1	1	1
0.6	0	0.2	0.3	0.5	0.7	0.8	1	1	1	1	1
0.7	0	0.1	0.3	0.4	0.6	0.7	0.9	1	1	1	1
0.8	0	0.1	0.3	0.4	0.5	0.6	0.8	0.9	1	1	1
0.9	0	0.1	0.2	0.3	0.4	0.6	0.7	0.8	0.9	1	1
1	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1

Proposition 2. *S_{G43} satisfies the Y axioms and does not satisfy the SD axioms.*

The fuzzy entropy measure is

$$\begin{aligned} E_{\mathrm{G43}}(A) &= S_{\mathrm{G43}}(A \cup \bar{A}, A \cap \bar{A}) \\ &= 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_{A \cup \bar{A}}(x) \neq 0}} \max(0, 1 - \frac{\mu_{A \cap \bar{A}}(x)}{\mu_{A \cup \bar{A}}(x)}) \\ &= 1 - \frac{1}{|X|} \sum_{x \in X} (1 - \frac{\min(\mu_A(x), 1 - \mu_A(x))}{\max(\mu_A(x), 1 - \mu_A(x))} \\ &= \frac{1}{|X|} \sum_{x \in X} \frac{\min(\mu_A(x), 1 - \mu_A(x))}{\max(\mu_A(x), 1 - \mu_A(x))} \ . \end{aligned} \tag{14}$$

Note the similarity of this entropy with the one derived by Kosko from his measure of subsethood

$$E_{\mathrm{K}}(A) = \frac{\sum_{x\in X} \min(\mu_A(x), 1-\mu_A(x))}{\sum_{x\in X} \max(\mu_A(x), 1-\mu_A(x))} \ . \tag{15}$$

3.3 The Modified Goguen Operator

Cited in [10], the rounded rational values of this operator in V_{11}^2 are shown in Table 4. Here, the violation of inclusion $A \subset B$ with regard to $x \in X$ is given by $\max(0, 1 - \frac{\mu_B(x)}{\mu_A(x)}, \frac{\mu_A(x)-\mu_B(x)}{1-\mu_B(x)})$ where $\mu_A(x) > 0$ and $\mu_B(x) < 1$. So the measure is

$$S_{\mathrm{mG43}}(A,B) = 1 - \frac{1}{|X|} \sum_{\substack{x\in X \\ \mu_A(x)>0 \\ \mu_B(x)<1}} \max(0, 1 - \frac{\mu_B(x)}{\mu_A(x)}, \frac{\mu_A(x)-\mu_B(x)}{1-\mu_B(x)}) \ . \tag{16}$$

Table 4. mG43 operator for value set V_{11}

mG43	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1	1	1	1	1	1	1	1	1	1	1
0.1	0	1	1	1	1	1	1	1	1	1	1
0.2	0	0.5	1	1	1	1	1	1	1	1	1
0.3	0	0.3	0.7	1	1	1	1	1	1	1	1
0.4	0	0.3	0.5	0.8	1	1	1	1	1	1	1
0.5	0	0.2	0.4	0.6	0.8	1	1	1	1	1	1
0.6	0	0.2	0.3	0.5	0.7	0.8	1	1	1	1	1
0.7	0	0.1	0.3	0.4	0.5	0.6	0.8	1	1	1	1
0.8	0	0.1	0.3	0.3	0.3	0.4	0.5	0.7	1	1	1
0.9	0	0.1	0.1	0.1	0.2	0.2	0.3	0.3	0.5	1	1
1	0	0	0	0	0	0	0	0	0	0	1

Proposition 3. *S_{mG43} satisfies the Y axioms and does not satisfy the SD axioms.*

The fuzzy entropy measure is

$$\begin{aligned} E_{\mathrm{mG43}}(A) &= S_{\mathrm{mG43}}(A \cup \bar{A}, A \cap \bar{A}) \\ &= 1 - \frac{1}{|X|} \sum_{\substack{x\in X \\ \mu_{A\cup\bar{A}}(x)>0 \\ \mu_{A\cap\bar{A}}(x)<1}} \max(0, 1 - \frac{\mu_{A\cap\bar{A}}(x)}{\mu_{A\cup\bar{A}}(x)}, \frac{\mu_{A\cup\bar{A}}(x) - \mu_{A\cap\bar{A}(x)}}{1-\mu_{A\cap\bar{A}}(x)}) \quad (17) \\ &= 1 - \frac{1}{|X|} \sum_{x\in X} \frac{\max(\mu_A(x), 1-\mu_A(x)) - \min(\mu_A(x), 1-\mu_A(x))}{\max(\mu_A(x), 1-\mu_A(x))} \ . \end{aligned}$$

3.4 The Gödel Operator

Listed in [4] (see Table 5), this operator gives up the measure

$$S_{\mathrm{S}^*}(A,B) = 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_A(x) > \mu_B(x)}} (1 - \mu_B(x)) \ . \tag{18}$$

Table 5. S* operator for value set V_{11}

S*	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1	1	1	1	1	1	1	1	1	1	1
0.1	0	1	1	1	1	1	1	1	1	1	1
0.2	0	0.1	1	1	1	1	1	1	1	1	1
0.3	0	0.1	0.2	1	1	1	1	1	1	1	1
0.4	0	0.1	0.2	0.3	1	1	1	1	1	1	1
0.5	0	0.1	0.2	0.3	0.4	1	1	1	1	1	1
0.6	0	0.1	0.2	0.3	0.4	0.5	1	1	1	1	1
0.7	0	0.1	0.2	0.3	0.4	0.5	0.6	1	1	1	1
0.8	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	1	1	1
0.9	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	1	1
1	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1

Proposition 4. *S_{S^*} satisfies the Y axioms and does not satisfy the SD axioms.*

Note the rigidity of the measure S_{S^*}: in (Y3) the $\geq$ of the consequent can be replaced by $=$, meaning that, once fixed B, the different cardinalities of A_1 and A_2 are independent of the measures $S_{\mathrm{S}^*}(A_1, B) = S_{\mathrm{S}^*}(A_2, B)$.

The fuzzy entropy measure is

$$\begin{aligned} E_{\mathrm{S}^*}(A) &= S_{\mathrm{S}^*}(A \cup \bar{A}, A \cap \bar{A}) \\ &= 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_{A\cup\bar{A}}(x) > \mu_{A\cap\bar{A}}(x)}} (1 - \mu_{A\cap\bar{A}}(x)) \\ &= 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_A(x) \neq 0.5}} \max(\mu_A(x), 1 - \mu_A(x)) \ . \end{aligned} \tag{19}$$

We observe that $E_{\mathrm{S}^*}(A) = E_{\mathrm{W}}(A)$.

3.5 The Standard Strict Operator

Also listed in [4] (see Table 6), the fuzzy binary relation obtained is

$$S_{\mathrm{S}}(A,B) = 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_A(x) > \mu_B(x)}} 1 \ . \tag{20}$$

Table 6. S operator for value set V_{11}

S	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1
0	1	1	1	1	1	1	1	1	1	1	1
0.1	0	1	1	1	1	1	1	1	1	1	1
0.2	0	0	1	1	1	1	1	1	1	1	1
0.3	0	0	0	1	1	1	1	1	1	1	1
0.4	0	0	0	0	1	1	1	1	1	1	1
0.5	0	0	0	0	0	1	1	1	1	1	1
0.6	0	0	0	0	0	0	1	1	1	1	1
0.7	0	0	0	0	0	0	0	1	1	1	1
0.8	0	0	0	0	0	0	0	0	1	1	1
0.9	0	0	0	0	0	0	0	0	0	1	1
1	0	0	0	0	0	0	0	0	0	0	1

This relation does not satisfy (Y2): if A is a fuzzy set such that $0.5 < \mu_A(x) < 1, \forall x \in X$, then

$$S_{\mathrm{S}}(A, \bar{A}) = 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_A(x) > \mu_{\bar{A}}(x)}} 1 = 0 \ . \tag{21}$$

So, we state the following

Proposition 5. *S_S does not satisfy the Y axioms nor the SD axioms.*

The function

$$\begin{aligned} E_{\mathrm{S}}(A) &= S_{\mathrm{S}}(A \cup \bar{A}, A \cap \bar{A}) \\ &= 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_{A \cup \bar{A}}(x) > \mu_{A \cap \bar{A}}(x)}} 1 \\ &= 1 - \frac{1}{|X|} \sum_{\substack{x \in X \\ \mu_A(x) \neq 0.5}} 1 \end{aligned} \tag{22}$$

is not an entropy measure in the sense of De Luca and Termini. A fuzzy set A with $\mu_A(x) \neq 0.5$, $\forall x \in X$, has null entropy, in contradiction with their first axiom.

4 Conclusion

We have extended the class of subsethood measures applied to fuzzy sets. It is shown that four of the proposed measures satisfy Young's axioms and new entropy measures are derived.

References

1. Bandler, W., Kohout, L.: Fuzzy power sets and fuzzy implication operators. Fuzzy Sets and Systems 4 (1980) 13–30
2. De Luca, A., Termini, S.: A definition of a nonprobabilistic entropy in the setting of fuzzy sets theory. Inform. Control **20** (1972) 301–312
3. Dubois, D., Prade, H.: Fuzzy Sets and Systems: Theory and Applications. Academic Press, New York (1980)
4. Gaines, B.R.: Foundations of fuzzy reasoning. Int. J. Man-Mach. Stud. **8** (1976) 623–668
5. Goguen, J. A.: The logic of inexact concepts. Synthese **19** (1969) 325–373
6. Halmos, P. R.: Naive Set Theory. Van Nostrand, Princeton (1960)
7. Kleene, S. C.: On a notation for ordinal numbers. J. Symb. Logic **3** (1938) 150–155
8. Kosko, B.: Neural networks and fuzzy systems. Prentice Hall, Englewood Cliffs (1992)
9. Mamdani, E. H.: Application of fuzzy logic to approximate reasoning using linguistic systems. IEEE Trans. Comput. **26** (1977) 1182–1191
10. Mizumoto, M., Zimmermann, H. J.: Comparison of fuzzy reasoning methods. Fuzzy Sets and Systems **8** (1982) 253–283
11. Sánchez, E.: Inverses of fuzzy relations. Application to possibility distributions and medical diagnosis. Fuzzy Sets and Systems **2** (1979) 75–86
12. Sinha, D., Dougherty, E. R.: Fuzzification of set inclusion: Theory and applications. Fuzzy Sets and Systems **55** (1993) 15–42
13. Smets, Ph., Magrez, P.: Implication in fuzzy logic. Int. J. Approx. Reasoning **1** (1987) 327–347
14. Willmott, R.: Two fuzzier implication operators in the theory of fuzzy power sets. Fuzzy Sets and Systems 4 (1980) 31–36
15. Wu, W.: Fuzzy reasoning and fuzzy relational equations. Fuzzy Sets and Systems **20** (1986) 67–78
16. Yager, R. R.: An approach to inference in approximate reasoning. Int. J. Man–Mach. Stud. **13** (1980) 323–338
17. Young, V. R.: Fuzzy subsethood. Fuzzy Sets and Systems **77** (1996) 371–384
18. Zadeh, L.A.: Fuzzy sets. Inform. Control **8** (1965) 338–353

Object Oriented Stochastic Petri Net Simulator: A Generic Kernel for Heuristic Planning Tools

Miquel A. Piera, Antonio J. Gambin, Ramon Vilanova

Dept. d'Informatica. Fac. Ciencies. Universitat Autonoma de Barcelona.
08193 Bellaterra. Catalonia. Spain
miquel@sunaut.uab.es
antonio@sunaut.uab.es

Abstract. To produce a large number of different products in an efficient way, traditional industry production architectures have incorporated non-value-adding operations such as transporting, storing and inspecting to provide flexibility. These operations should be minimised, if not eliminated, to improve performance on automated manufacturing systems. Heuristic planning tools seem to offer a good methodology to deal with a sub-optimal scheduling policy of present flexible manufacturing systems.In this paper an approach to evaluate and improve heuristic planning tools results will be presented.

1 Introduction

World-wide market competition, high product quality requirements, together with random demands instead of steady demand, are some key-factors which have forced Industry to change both:

- Traditional rigid and/or non-automated *production architectures* (such as Flow Shop, Job Shop) towards flexible manufacturing systems (FMS) [4].
- Traditional *production planning methodology* towards new heuristic based methodologies which could cope with a large amount of decision variables inherent to present FMS architectures.

The exact optimal solution of a FMS planning problem is quite complex and difficult, may be impossible to obtain. Furthermore, when system behaviour is subject to random disturbances, system uncertainty should be described by the mathematical model (stochastic processes), which will increase considerably the complexity of the methodology to be implemented to determine the optimal solution.

Conventional planning tools that handled scheduling problems by using analytical techniques often fail to catch the appropiate level of detail when applied to FMS. For instance, Queuing theory methods can model the steady state operations, but they fall short to deal with transients. Hierarchical Planning Tools offer good results [3] to deal with complex problems which might be decomposed in "independent" subtasks (ie. short, medium and long term production planning). However, these tools are inadequated to solve internal FMS planning problems, where machine scheduling can not be treated as independent targets.

A flexible manufacturing system (FMS) is a production system consisting of a set of identical and/or complementary numerically controlled machines which are connected through an automated transportation system. In addittion, each process in a FMS is controlled automatically by a dedicated computer. Under *ideal* operating conditions, a FMS is capable of processing workpieces of a certain workpiece-spectrum in an arbitrary sequence with negligible setup delays between operations. However, setup delays (*real operating conditions*) can decrease FMS performance results drastically if decission variables, such as processing, handling, storing and transportation, are not well coordinated. Consider for example an erroneous batch size when FMS processing units must process different short duration time tasks.

Intrinsic FMS characteristics which constrains the use of some traditional production planning techniques are:

- Uncertainty in demand and time production : FMS production and transport units behaves as discrete event oriented systems.
- Large decision variables amount. Note that while *flexibility* is essential to competitiveness, the amount of decission variables which should be coordinated and synchronized in an efficient way is a major drawback.
- Quick solutions to react in front perturbations: Most optimal planning techniques are C.P.U. intensive (time consuming) which make them unsuitable to be used for re-scheduling purposes.

Furthermore, non-value-added operations (such as transporting, storing and inspecting) incorporated in manufacturing systems to allow a higher flexibility level, are the main operations which should be minimised, if not eliminated by the planning tool. These opposed goals force to look for new planning methodologies which could lead with a good weighted compromise betwen different economic objectives to be satisfied.

The use of effective search algorithms for selecting candidate schedules are essential to deal with the exponential growth of possible schedules. Digital simulation techniques are a very useful platform to deal with a FMS schedule which could contain all the necessary details to instruct every machine so that the schedule could be executed.

Unfortunately, most of commercial discrete event oriented simulation packages are designed to perform simulation as an analysis tool. That is, the system to be studied is modelled, perturbed in some interesting fashion, and simulated to predict what changes that disturbances would cause in a real system.

A successful plan depends on the model structure used to describe the system, and the ability to break up a complex goal into a series of finite ordered sequence of steps. This task decomposition into a sequence of steps is common to most planning approaches, however, a good planner is characterised by the ability to represent this decomposition in a compact and effective manner, and then search over many different sequences of tasks execution to find plans which will both successfully reach the goal and may provide performance benefits such as short time or low cost.

To deal with such a planner, an Object Oriented Stochastic Petri Net Simulation kernel which could be used as a generic planning tool to cope with the scheduling and routing problems of a FMS has been developped. Section II introduces the needs to describe the logic constraints which appear between FMS units, and the advantges of using Petri Nets as a tool to formalize the model. Section III presents a time consuming methodology to determine the optimal plan. Finally,section IV summarizes the benefits of the implemented tool to evaluate and improve heuristic planning tools.

2 Modelling FMS by Stochastic Timed Petri Net

Petri Nets (P.N.) have shown to be successful tools for modeling FMS's [2] due to several advantages such as its graphical nature, the conciseness of embodying both the static structure and the dynamics, and the availability of the mathematical analysis techniques. Furthermore, P.N. are very suitable to model and visualise patterns of behaviour comprising, concurrency, synchronisation and resource sharing, which are the main characteristics of a FMS.

To describe the logic constraints between a list of resources (processing machines, transport units, local stocks) a list of operations, and their precedence relationships, a P.N. model might suffice. However, to deal with good coordination of all these resources, the independent variable time should also be modelled.

Numerous tools have been added to Petri Nets by the research community to enable more condensed descriptions where the time factor takes part: Synchronized P.N., Timed P.N., Stochastic P.N.,Coloured P.N., Continuous P.N., and others.

A Timed Petri Net enables a system the functioning of which is time dependent to be described. Timed P.N. can be used as a good tool to evaluate the performance of a system. Two main methods exist for modelling timing: the timings are either associated with the places (the PN is said to be P-timed), or with the transitions (the PN is said to be T-timed).

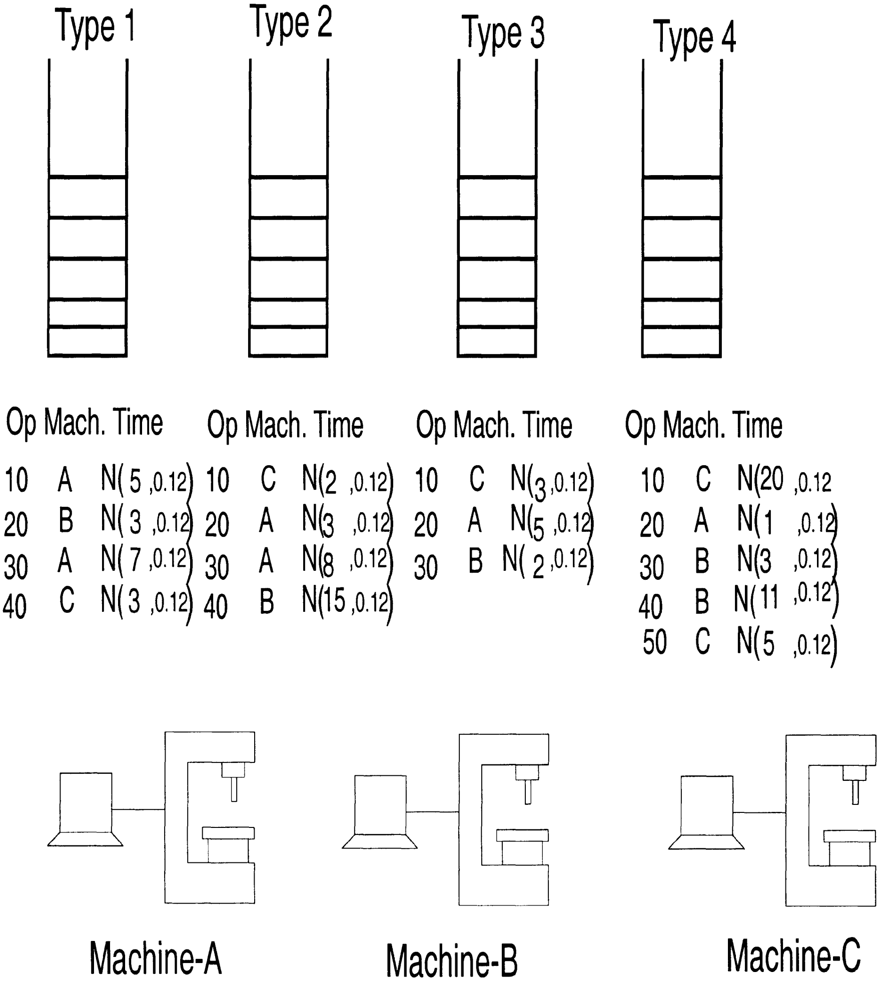
When the duration of a firing transition behaves as a stochastic process, systems are modeled by Stochastic T-timed Petri Nets. This is the model structure that we have chosen to model FMS where machine working time and transport time are described by random variables.

2.1 Example:

A small factory has been chosen to illustrate some usual FMS production planning characteristics, such as:

- A FMS production planning method should allow for various hierarchical decision-making levels, from simple deterministic conditional branches to multicriteria, multiconditional branches.

- The proposed system provides a source for evaluating several alternate designs, and correct the heuristic kernel to choose a desired sequence of transitions.
- The problem shows a dynamic structure which allows for the investigation of several different decision strategies.



Op	Mach.	Time	Op	Mach.	Time	Op	Mach.	Time	Op	Mach.	Time
10	A	N(5,0.12)	10	C	N(2,0.12)	10	C	N(3,0.12)	10	C	N(20,0.12
20	B	N(3,0.12)	20	A	N(3,0.12)	20	A	N(5,0.12)	20	A	N(1,0.12)
30	A	N(7,0.12)	30	A	N(8,0.12)	30	B	N(2,0.12)	30	B	N(3,0.12)
40	C	N(3,0.12)	40	B	N(15,0.12)				40	B	N(11,0.12)
									50	C	N(5,0.12)

Fig. 1. A Flexible Manufacturing System

Figure 1 shows the FMS resources (Machine A,B,C) together with the sequence of operations to process each unit type (4 different unit types) and the time required to finish each operation, and the machine where the operation should be made. Note that Machines A,B and C are reprogrammable shares resources that should be scheduled in an efficient way to minimize setup and processing times.

Constraints to be considered when sequencing operations are:

- Total inventory inside the FMS (W.I.P.) may not exceed a fixed amount of 50 pieces of each type.
- Once a set of 4 parts (one of each Part type) reaches the assembly point, they are assembled and inmediately removed from the total inventory.
- A set-up time of 5 minutes occurs whenever a Machine is switched from one mode of operation to another.

Figure 2 shows 4 subpetri nets which correspond to each machine model. The Petri Net of the overall system is built by merging places with the same name, and adding arcs to transitions with the same name. Thus, preconditions associated to transition T_{13} are places P_{12},P_{22}, P_{27} and P_{34}, which correspond to have available at least one piece of each part completely processed before the assembly process (T_{13}) could be activated.

A plan for this system will consist to define an ordered sequence of transitions (or actions) in such a way that firing this sequence will allow to drive the system from its initial state to a certain final state. The best heuristic planning tool will be the one that will reach the final state in a minimum time. Note that the time to drive the system from an initial state to the final state can not be computed by adding the time associated to each transition described by the plan, since some transitions can be fired in parallel [5].

2.2 The Coverability Tree

The coverability tree [1] allows to describe both:

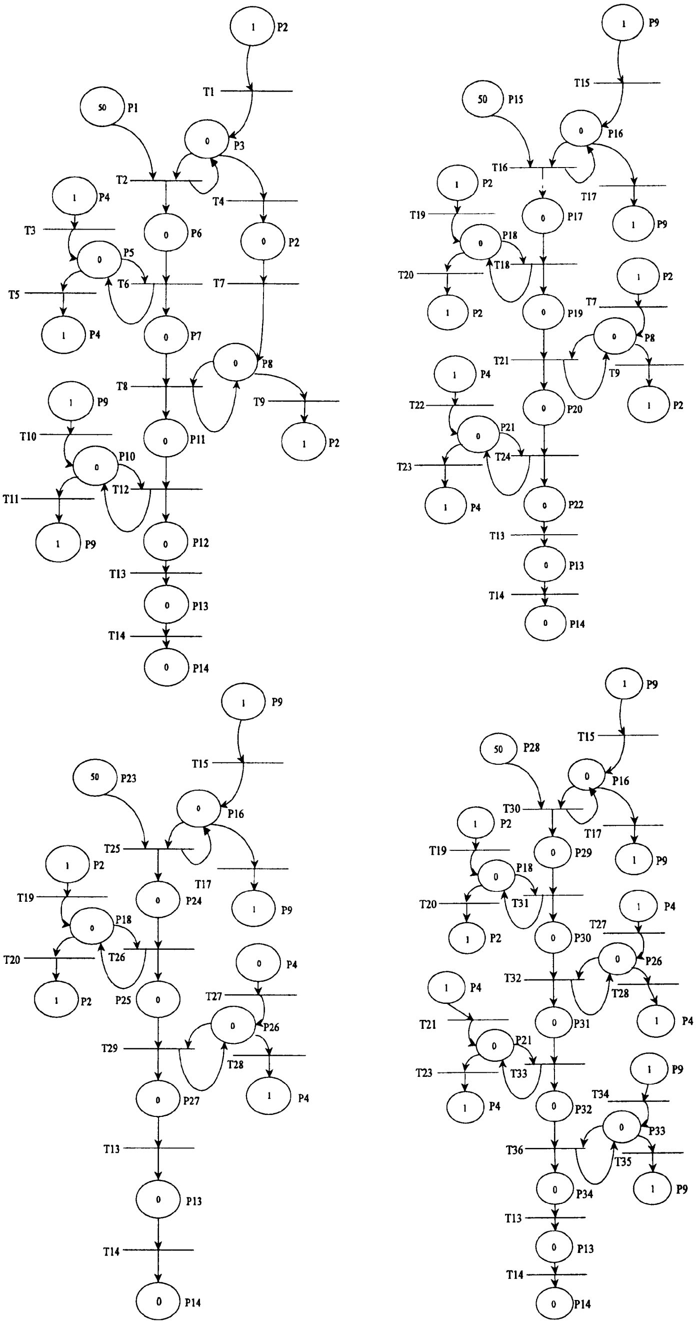
- All the FMS states (markings) which can be reached starting from certain initial system conditions M_0
- The transition sequence to be fired to drive the system from a certain initial condition to a desired end-state.

The main disadvantage of Petri Nets lies in the size of marked graphs (coverability tree) produced by modelling very complex discrete event systems, such as flexible manufacturing systems. For these large nets, although analytic techniques are available for their processement, the computation involved in analysing them is quite substantial, and in many cases, it may not be practical.Figure 3 illustrates this problem.

The initial state (M_0) of the coverability tree is represented by the vector:

$$M_0 = ([50,1,0,1,0,0,0,0,1,0,0,0,0,0,50,0,0,0,0,0,0,0,50,0,0,0,0,50,0,0,0,0,0,0,0])$$

which states that the three machines are in setup ($P_2 = 1$,$P_9 = 1$,$P_4 = 1$) and 50 pieces in each stock ($P_1 = 50$, $P_{15} = 50$, $P_{23} = 50$, $P_{28} = 50$). The scheduling goal will consist to find the sequence of operations that will allow to



TRANSITIONS LIST

T1: Setup to Idle Machine A Operation 10
T2: Machine A Operation 10 Part1 Executing
T3: Setup to Idle Machine B Operation 20
T4: Idle Machine A Operation 10 to setup
T5: Idle Machine B Operation 20 to setup
T6: Machine B Operation 20 Part1 Executing
T7: Setup to Idle Machine A Operation 30
T8: Machine A Operation 30 Part1 Executing
T9: Idle Machine A Operation 30 to setup
T10: Setup to Idle Machine C Operation 40
T11: Idle Machine C Operation 40 to setup
T12: Machine B Operation 40 Part1 Executing
T13: Assembly process
T14: Exiting product process
T15: Setup to Idle Machine C Operation 10
T16: Machine C Operation 10 Part2 Executing
T17: Idle Machine C Operation 10 to setup
T18: Machine A Operation 20 Part2 Executing
T19: Setup to Idle Machine A Operation 20
T20: Idle Machine A Operation 20 to setup
T21: Machine A Operation 30 Part2 Executing
T22: Setup to Idle Machine B Operation 40
T23: Idle Machine B Operation 40 to setup
T24: Machine B Operation 40 Part2 Executing
T25: Machine C Operation 10 Part3 Executing
T26: Machine A Operation 20 Part3 Executing
T27: Setup to Idle Machine B Operation 30
T28: Idle Machine B Operation 30 to setup
T29: Machine B Operation 30 Part3 Executing
T30: Machine C Operation 10 Part4 Executing
T31: Machine A Operation 20 Part4 Executing
T32: Machine B Operation 30 Part4 Executing
T33: Machine B Operation 40 Part4 Executing
T34: Setup to Idle Machine C Operation 50
T35: Idle Machine C Operation 50 to setup
T36: Machine C Oberation 50 Part4 Executing

PLACES LIST

P1: Stock Parts 1
P2: Machine A Setup
P3: Machine A Operation 10 Idle
P4: Machine B Setup
P5: Machine B Operation 20 Idle
P6: Machine A Part 1 Operation 10 Finished
P7: Machine B Part 1 Operation 20 Finished
P8: Machine A Operation 30 Idle
P9: Machine C Setup
P10:Machine C Operation 40 Idle
P11: Machine A Part 1 Operation 30 Finished
P12: Machine C Part 1 Operation 40 Finished
P13: Product Assembled
P14: System Exit
P15: Stock Parts 2
P16: Machine C Operation 10 Idle
P17: Machine C Part 2 Operation 10 Finished
P18: Machine A Operation 20 Idle
P19: Machine A Part 2 Operation 20 Finished
P20: Machine A Part 2 Operation 30 Finished
P21: Machine B Operation 40 Idle
P22: Machine B Part 2 Operation 40 Finished
P23: Stock Parts 3
P24: Machine C Part 3 Operation 10 Finished
P25: Machine A Part 3 Operation 20 Finished
P26: Machine B Operation 30 Idle
P27: Machine B Part 3 Operation 30 Finished
P28: Stock Parts 4
P29: Machine C Part 4 Operation 10 Finished
P30: Machine A Part 4 Operation 20 Finished
P31: Machine B Part 4 Operation 30 Finished
P32: Machine B Part 4 Operation 40 Finished
P33: Machine C Operation 50 Idle
P34: Machine C Part 4 Oberation 50 Finished

Fig. 2. F.M.S. Petri Net Model

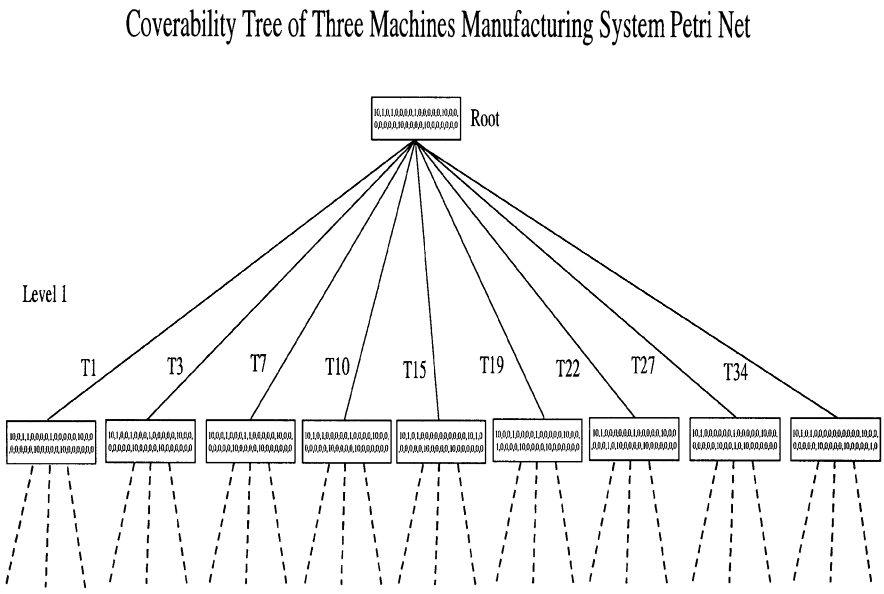


Fig. 3. First Coverability Tree level

drive the system from its original state to the final state described by the vector:

$$M_f = ([0,1,0,1,0,0,0,0,1,0,0,0,50,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0,0]).$$

Since it will not always be possible to build the overall coverability tree, two solutions can be adopted:

- Choose a higher abstraction level to describe FMS particular characteristics.
- Use some heuristic to prune the tree.

The former option can reduce the size of the coverability tree considerably, however, it is not possible to afford a plan with appropiate detail to generate a planning that could contemplate all the production architecture decision variables. Although the latter option does not guarantee that the optimal solution will be found, it offers a good answer to Industry scheduling and re-scheduling requirements which are forced (due to time constraints) to accept a prompt quick sub-optimal solution rather than a delayed optimal solution. Note that a fast re-scheduling is essential to quickly react when disturbances appear.

3 A Tool to Determine the Optimal Plan

A coverability tree of a FMS can rise in size in an exponential way with respect to the number op units to be processed. Due to computer restrictions, is not possible to build, analyze and maintain the overall coverability tree in a computer memory.

An algorithm has been implemented [5] to go over the an entire coverability tree just by storing in memory a static structure describing the FMS Petri Net model, a dynamic list with the trasition sequence fired to get the present state, a static structure storing information associated to tokens, and a file storing coded information describing the marking generated when building the coverability tree.

The algorithm begins with an initial marking, and computes all the activated trasitions for this state. A transition is selected, a new marking is computed, and checked for its existence in a file. If the a new mark was generated previously at some other level of the tree (old node),the algorithm do not explore the activated transitions associated to the new state. However, if the computed marking correspond to a new state not previously computed, the transition is added to a dynamic list, the new state is codified and added to the file together with some extra information, and all the activated transitions from this new state are computed and processed in a similar iterative way again. When a computed marking agrees with the goal state, the current transition list together with its processing time is stored in a file.

Note, that a marking vector together with the last transition fired is enough to backward to the previous marking vector, and analyse another marking by selecting another activated transition. Thus, it is possible to generate the entire coverability tree of a system and determine the best sequence of actions to drive the FMS from an initial state to a goal state.

This methodology to determine an optimal plan is time consuming (several days) and useless to be used to be used on-line to rule a FMS. However, it gives very good results to evaluate and correct heuristic based planners which can be used later on to obtain fast scheduling and re-scheduling FMS solutions.

4 A Kernel for Heuristic Planning Tools Performance Evaluation

Figure 4 shows the main elements used by the object oriented stochastic petri net implemented. The main characteristics of these elements are:

- FMS Model: The Petri net model is described sintactically in an ASCII file. This file specifies the weight of the arcs between the Places and the Transitions of the P.N., together with the time assigned to each transition.
- Cost Function: To evaluate the price to get a certain state. An user can specify an analytical cost function as a mathematical expression of places values and time.
- Scheduling Target: The goal is specified by the final state of the P.N. The initial conditions of the FMS are specified on the same file.
- Heuristics: An end user can specify in an ASCII file a set of rules to be used by the simulator to prune the coverability tree in a certain way.

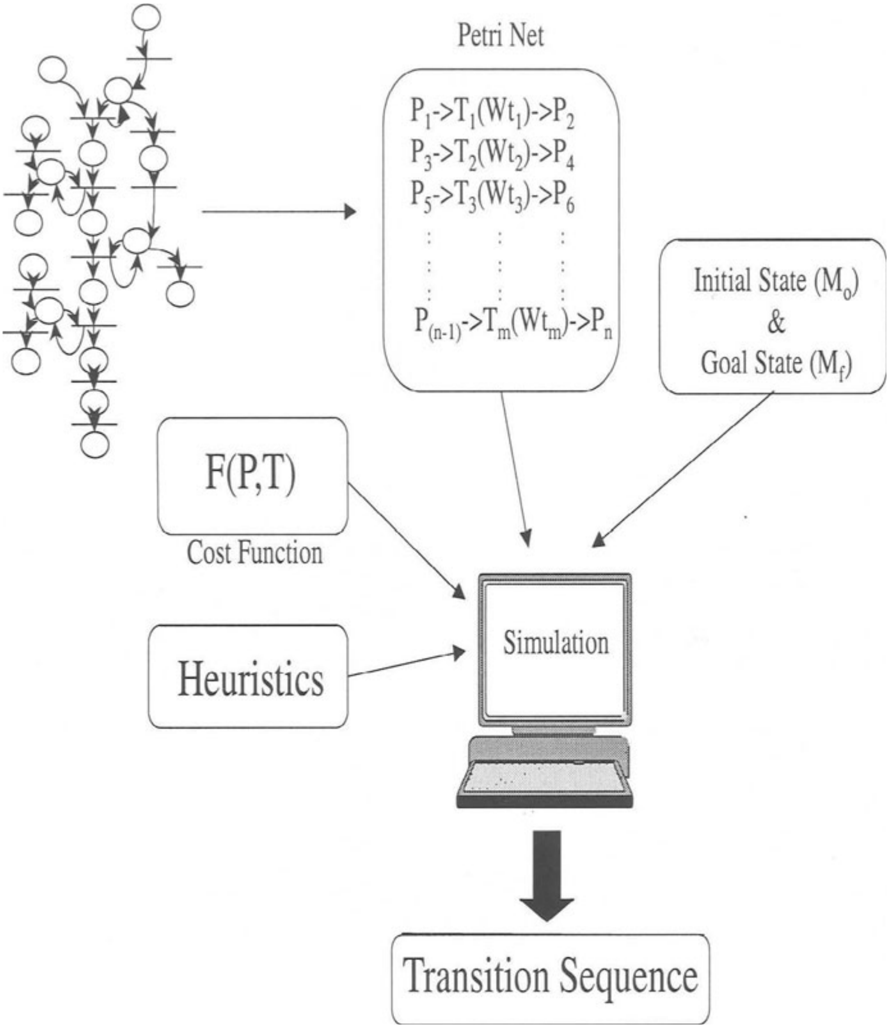

Petri Net
P_1->T_1(Wt_1)->P_2
P_3->T_2(Wt_2)->P_4
P_5->T_3(Wt_3)->P_6
$P_{(n-1)}$->T_m(Wt_m)->P_n
Initial State (M_0)
&
Goal State (M_f)
F(P,T)
Cost Function
Heuristics
Simulation
Transition Sequence

Fig. 4. Main Elements of the Simulator Prototype Implemented

- Simulator: The software package implemented, analyses the P.N. described in the FMS Model file, and prune the coverability tree according to the cost function specified by the user together with the rules specified in the heuristic file. As a result, the simulator will provide the sequence obtained to get the goal state from the initial state.

5 Conclusions

An Object Oriented Stochastic Petri Net Simulator has been developed to generate a schedule policy in order to satisfy certain production goals on a flexible manufacturing system. The proposed framework allows to evaluate different heuristic rules to drive a Petri Net simulator from an initial state to a desired goal state in an interactive way.

6 Acknowledgements

The authors wish to record their appreciation for the financial support received from the CICYT Spanish program TAP95-0830.

References

[1] J-M. Proth, X. Xie.: Petri Nets. A Tool for Design and Management of Manufacturing Systems. Wiley,1996.

[2] M. Silva and R. Valette: Petri nets and flexible manufacturing. LNCS: Advances in Petri Nets, pp.374-417. 1989

[3] Suresh P. Sethi: Hierarchical Decision Making in Stochastic Manufacturing Systems .S.C.1994.

[4] H. Tempelmeier H. Kuhn. Flexible Manufacturing Systems. Wiley,1993.

[5] A.J. Gambin. Analisis del arbol de cobertura. Internal report. 1997.

An Artificial Neural Network for Flashover Prediction. A Preliminary Study

Christian W Dawson[1], Paul D Wilson[1], Alan N Beard[2]

[1] *School of Mathematics and Computing, University of Derby, Kedleston Road,Derby, DE22 1GB, UK*

[2] *Department of Civil and Offshore Engineering,Heriot-Watt University,Edinburgh, EH14 4AS, UK*

Abstract. Trying to estimate the probability of a flashover occurring during a compartment fire is a complex problem as flashovers depend on a large number of factors (for example, room size, air flow etc.). Artificial neural networks appear well suited to problems of this nature as they can be trained to understand the explicit and inexplicit factors that might cause flashover. For this reason, artificial neural networks were investigated as a potential tool for predicting flashovers in a room with known, or estimable, compartment characteristics. In order to deal with uncertainties that can exist in a model's results, a statistical analysis was employed to identify confidence intervals for predicted flashover probabilities. In addition, Monte Carlo simulation of trained artificial neural networks was also employed to deal with uncertainties in initial room characteristic estimates. This paper discusses these analyses and comments on the results that were obtained when artificial neural networks were developed, trained and tested on the data supplied.

1. Introduction

Fire growth in an enclosure is a non-linear process characterised by a complex interaction of growth factors. Threat to life and property increases dramatically if a sudden transition from localised burning to full room involvement occurs. Such a transition is known as *flashover* and may occur with a certain probability depending on fire growth factors (Beard et alia 1993).

The problem is to classify compartment fires as either likely to result in flashover or, alternatively, to burn to extinction in a more stable fashion. Difficulties of deriving deterministic models for this phenomenon are well known (Beard et alia 1993) and have led to this study of the classification problem using artificial neural networks (ANNs).

Although advances have been made in recent years in fire modelling (for example, Peacock et alia 1991; Moore 1995) the prediction and modelling of flashover is an area that has yet to be fully explored (see, for example, Beard et alia 1994). This paper describes the application of artificial neural networks to the problem of predicting flashover in compartment fires and discusses how uncertainties can be accommodated by ANN simulation and statistical analyses.

2. Artificial Neural Networks

2.1 Overview

Artificial neural network research tends to fall into two main categories; first is the development and improvement of existing ANN training algorithms and topologies. The second area involves the application and evaluation of existing ANN research to complex problems. This is the category into which this particular study falls. Existing ANN algorithms have been adopted and adapted to the problem of flashover prediction. In this case the networks chosen for this study were feed forward, three-layered networks, trained using a modified backpropagation algorithm.

The three layers in this case are an *input* layer, which distributes inputs to the *hidden* layer, and an *output* layer. The number of nodes used in the input layer and output layer are determined by the problem to which the network is being applied. The number of nodes used in the hidden layer is chosen by the user and there are no clear rules as to how many nodes one should employ in this layer (although suggestions have been made; Lippmann, 1987). Consequently, different numbers of hidden nodes were evaluated in this study in an attempt to identify the 'best' ANN configuration for flashover prediction.

Explanation of ANN structures and discussion of the backpropogation training algorithm is beyond the intended scope of this paper as there are numerous articles and texts devoted to these issues. For more information on ANNs the reader is directed towards texts such as Lippmann (1987), Wasserman (1989) and Gallant (1994).

In order to implement and evaluate ANNs, four computer programs were developed in this study. The first program was used to generate training sets from the flashover data that were available. The second program standardises these data before they are passed on to a third program that generates and trains the ANNs. The backpropogation algorithm used for training the ANNs in this program was modified to include *momentum* (Gallant 1994) and a dynamically controlled step size algorithm (Dawson 1996). The fourth program takes a trained neural network and allows a user to test its responses by asking it various questions. These questions can include Monte Carlo simulation of the network that provide estimates which are based on initial best guesses by the user. The distribution functions generated by this program, as text files, can then be analysed to determine the uncertainties involved with the estimated probabilities (Dawson 1996) - see below.

2.2 Neural Network Simulation

Artificial neural networks need not be limited to deterministic inputs that produce a single output value from a single set of input stimuli. Multiple outputs, that provide some indication of variability based on 'fuzzy' inputs, can be achieved by simulating trained networks using a Monte Carlo approach. In this case, the trained networks provide probability distribution functions representing the uncertainty involved in predicting a potential flashover. Figure 1. provides a visual overview of this concept.

Quite clearly, if the initial inputs are best guesses, any variations from an accurate value can have a significant affect on the value of a trained network's response. A network should be able to provide an indication of the uncertainty involved in its outputs, knowing the variability of the initial inputs.

In order to achieve this aim, the networks developed in this study can be simulated with selected input distribution functions; such as normal, rectangular, and triangular. Thus, if a user provides an initial estimate of a room's fire loading (a measure of the amount of combustible material in a room) as, for example, 50 units plus or minus 20%, a trained network can simulate this input as a rectangular distribution with limits 40 to 60. The network samples from this distribution a number of times (selected by the user, but usually about 10000) and builds up a picture of the likely probability of flashover based on this fuzzy input. In Figure 1, a fuzzy input is represented by a normal distribution function. The output then generated by the simulation in this figure is represented by the flashover probability distribution.

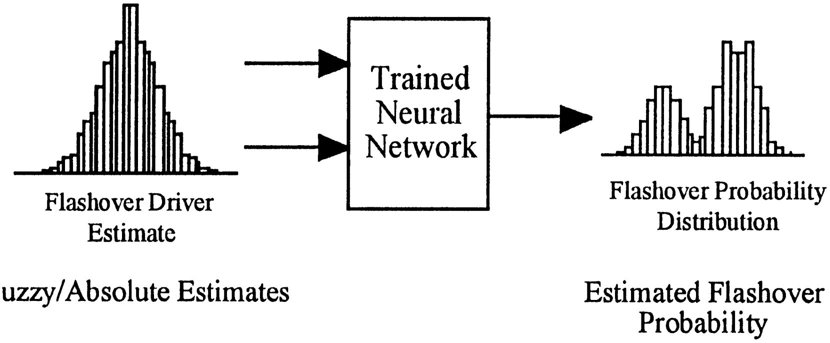


Fig. 1. Overview of Simulated Networks for Flashover Probability Estimation

3. Neural Network Training

3.1 Overview

As with the development of any model, one is always limited by the quantity and quality of data that are available. In this case, the quality of data depended on expert judgment and the quantity of data supplied was somewhat limited. However, there were sufficient data available for this preliminary study. The data consisted of fourteen flashover probability values based on three factors (*drivers*); Room Size, Fire Load, and Opening Factor (a measure of ventilation). These three factors were used as the three inputs into an ANN with probability of flashover being used as the output.

3.2 Neural Network Training

There are a number of different parameters one can manipulate when constructing ANNs. These parameters include variations to the number of hidden nodes, variations to the 'speed' of the training algorithm employed (by adjusting a *learning parameter*), and variations to the length of time one chooses to train a network for (the number of training cycles or *epochs*). In this particular study a number of different network configurations were investigated (different numbers of hidden nodes) along with various training cycles. The results of the more accurate configurations are presented later.

To test the ANNs in this study the data had to be split into training sets and calibration sets. As there were only fourteen data points available it was decided that networks would be trained on thirteen of these data points and tested on the fourteenth - ie the data point the networks had not seen before. This procedure was repeated fourteen times, changing the unseen data point each time, so that all data points were predicted by the ANNs at some stage. This approach was also used to evaluate the comparative statistical model which is discussed below.

4. Evaluation

4.1 Comparative Statistical Model

In order to evaluate the effectiveness of the ANN approach in this study, results were compared with a more conventional statistical model; stepwise multiple linear regression (MLR) in this case. This technique identified the most significant factors affecting flashover from the three available and reduced the affects of multicollinearity between these factors by removing closely correlated ones from the model. In this study it was found that there was a close correlation between Opening Factor and Fire Load and consequently Opening Factor was excluded from the final linear regression model.

4.2 Measuring Accuracy

There are a number of measures one can use to determine the overall accuracy of modelled results. In this study two such measures were employed; the mean squared error (MSE) and the mean squared relative error (MSRE):

$$\mathrm{MSE} = \frac{\sum_{i=1}^{n} (P_i - \hat{P}_i)^2}{n} \qquad \mathrm{MSRE} = \frac{\sum_{i=1}^{n} \frac{(P_i - \hat{P}_i)^2}{P_i}}{n}$$

where $\hat{P}_i$ are the n modelled probabilities and P_i are the n actual probabilities.

Squared errors provide a good indication of errors made at high probabilities, whilst relative errors provide a balanced perspective of the goodness of fit for low probabilities (Karunanithi et alia 1994). In addition, the coefficient of determination, r^2, was calculated for each model to measure how successfully the predicted probabilities followed actual probabilities. Whilst the MSE and MSRE provide an indication of the 'absolute' errors made by the models, the r^2 statistic provides a useful indication of whether a model follows general increases and decreases in probability fluctuations.

4.3 Results

Table 1 summarises the results that were obtained during this study. The results presented are for the step wise multiple linear regression model and an ANN with 8 hidden nodes trained for 5000 epochs. The table presents the coefficient of determination, r^2, for each model and the MSE and MSRE.

Model	r^2	MSE	MSRE
ANN	0.913	0.0074	1.493
SWMLR	0.941	0.0052	0.14

Table 1 Summary of Results from SWMLR and ANN Models

In this table it clear that the SWMLR approach provides more accurate results than the ANN model produced. However, this is not surprising as a SWMLR model should capture, quite accurately, any linear relationship from such a limited number of data points. These results should not detract from those of the ANN approach which does appear very accurate overall. The ANN model follows the actual results trend very well from the coefficient of determination and the MSE shows a good absolute accuracy from the prediction. Perhaps the only disappointment is the high MSRE which does indicate that the ANN model is producing poor predictions at lower probabilities than it could do.

Figure 2 provides a useful visual comparison of both model's results. In this case *actual probability* is plotted against *modelled probability*. An exact model would result in all points lying on the line f(x) = x; shown by the bold line in these figures. However, although this is not achieved, both models do show reasonable accuracy over the entire range of their predictions. Using the results of these analyses in this form (ie regressing *actual probability* against *modelled probability*) allows the confidence limits of a model's results to be calculated as discussed below.

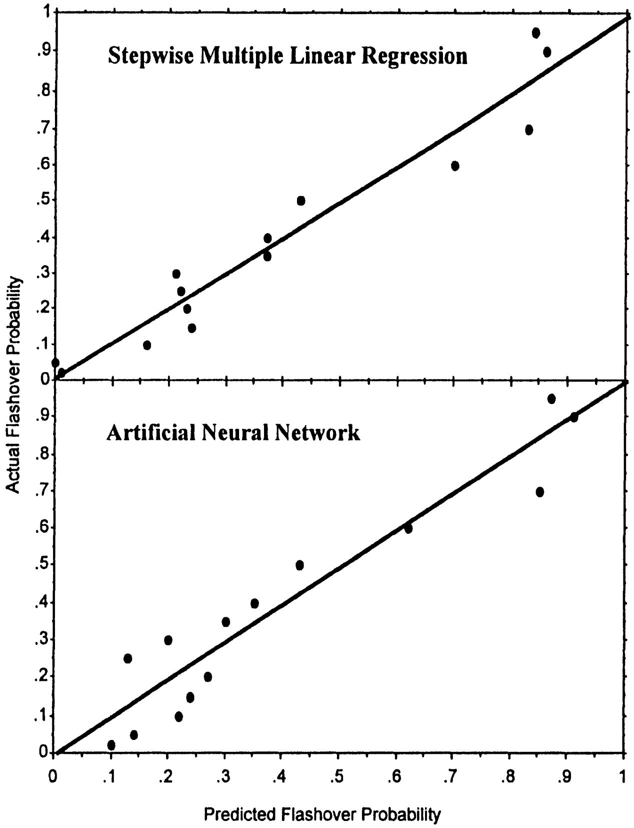


Figure 2 Comparative Results of an ANN and Statistical Model

4.4 Confidence Limits

A useful outcome from the above evaluation was determining the confidence one has in a model's predictive powers. Using the standard error of the least squares regression line of actual duration on predicted duration, one can provide confidence intervals for each model's predictions. For example, in the ANN model, the standard error of the probabilities is 0.093. Thus, for an ANN flashover probability prediction of 0.5, one is 68% confident (one standard error about the regression line) that the actual probability of flashover the ANN model is attempting to predict lies in the range [0.395, 0.581] (based on an observed probability from the regression line being centred around 0.488). However, in order for this estimate to be valid, these results must be homoscedastic. Unfortunately, this is difficult to determine although a visual inspection of these data does indicate that homoscedasticity is preserved in this particular case.

4.5 Monte Carlo Simulation

As discussed earlier, in certain circumstances it is difficult to accurately predict the fire characteristics of a compartment. In these cases one must make subjective estimates of a compartment's fire load and opening factor. These estimates can be represented by probability distribution functions as the following example shows.

In this example the room is known to be 100m^3 but the fire load can only be estimated as somewhere between 40 and 60 units and the opening factor as between 1 and 2 units. Using Monte Carlo simulation of the trained ANN for 10000 iterations, the probability distribution of flashover probability is generated and is shown in figure 3. In this example, the mean probability of flashover is 0.625 with a standard deviation of 0.087. The minimum probability is 0.45 and the maximum is 0.79. Although this range is somewhat broad, its information content is significantly greater than a single probability value. In this case, with a fire load of 50 units and an opening factor of 1.5 units, the ANN predicts the probability of flashover as 0.625; equal to the mean of the simulated values. However, this deterministic prediction gives no indication of the uncertainty involved as, for example, one is really only 68% confident that the mean probability of flashover in this room lies in the range [0.54, 0.71]. The risk of flashover could be as high as 0.79 or as low as 0.34 in this case.

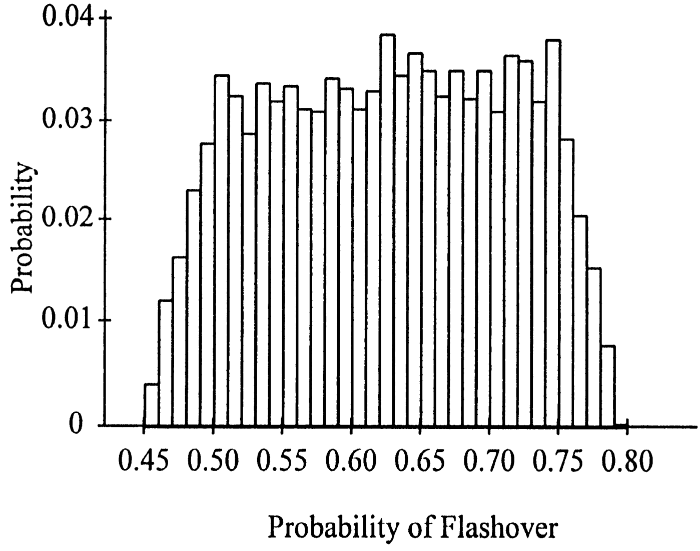


Figure 3 Example Probability Distribution from a Simulated ANN

5. Conclusions

This note has shown that an ANN model can provide reasonable estimates of the probability of flashover occurring in burning rooms. Although a stepwise multiple linear regression model provided slightly better results, the attractiveness of an ANN model is undiminished. Such a model could be adapted to include additional explicit and inexplicit factors with relative ease and could be strengthened with additional training data as and when they became available.

Analyses of the models also provides supplementary practical information. In this case, confidence intervals for each model's predictions were calculated. However, unless the accuracy of a model is reasonable, and predicted results are relatively homoscedastic, the results a model produces can be so vague as to be meaningless.

Clearly, further research is required to determine an optimum ANN configuration (topology, training rate, epochs and so on) for problems of this nature. As with all such models, improvements can continue to be made as more data become available and ANNs are able to learn from a larger historical data pool.

6. References

Beard, A.N. Bishop, S.R. Drysdale D.D. and Holborn P.G. (1993): Nonlinear Dynamics of Flashover in Compartment Fires. Fire Safety Journal, Vol 21, pp 11-45

Beard, A.N. Drysdale, D.D. Holborn, P.G. and Bishop, S.R. (1994): A Model of Instability and Flashover. Journal of Applied Fire Science, Vol 4(1), pp 3 - 16

Dawson, C.W. (1996): A Neural Network Approach to Software Project Effort Estimation. Applications of Artificial Intelligence in Engineering XI, Eds Adey, R.A., Rzevski, G. and Sunol, A.K., Computational Mechanics Publications, UK, ISBN: 1-85312-410-9, pp 229 - 237

Gallant, S.I. (1994): *Neural Network Learning and Expert Systems.* Massachusetts Institute of Technology, USA, 1994

Karunanithi, N. Grenney, W.J. Whitley, D. and Bovee, K. (1994): Neural Networks for River Flow Prediction. Journal of Computing in Civil Engineering, Vol 8(12), pp 201 - 220

Lippmann, R.P. (1987): An Introduction to Computing with Neural Nets. IEEE ASSP Magazine, April, pp 4 - 22

Moore, P. (1995): Why Fires go to Blazes. New Scientist, June, pp 26 - 30

Peacock, R.D Sanford, D. and Babrauskas, V. (1991): Data for Room Fire Model Comparisons. Journal of Research of the National Institute of Standards and Technology, Vol 96, pp 411 - 462

Wasserman, P.D. (1989): Neural Computing Theory and Practice. Van Nostrand Reinhold, New York

Fuzzy Decision Making Under Uncertainty

Cengiz Kahraman

Istanbul Technical University
Department of Industrial Engineering
and
80680 Maçka-Istanbul-Turkey
E-mail: cengiz2 @ ayasofya.isl.itu.edu.tr
Home Page: http://members.tripod.com/~ecem/kahraman.html

Ethem Tolga

Galatasaray University
Faculty of Technology

Engineering
80840 Ortakoy-Istanbul
Turkey

Abstract: When no probabilities are available for states of nature, decisions are given under uncertainty. When probabilities are unattainable, the criteria such as minimax, maximin, minimax-regret can be used. While these criteria are used, a single value is assigned for every strategy and state of nature. Fuzzy numbers are a good tool for the operation research analyst facing uncertainty and subjectivity. A triangular fuzzy number has been used instead of a single value of outcome. Numerical Examples have been given for every fuzzy decision criterion.

Keywords: Fuzzy set theory, fuzzy number, decision criteria, uncertainty.

1. Introduction

When outcomes are presumed to be known, a decision is said to be made under a condition of certainty. Under conditions of risk and uncertainty, possible future states are identified, and outcomes are estimated for each alternative by assuming each state will indeed occur. The difference between risk and uncertainty depends on the capability of estimating the likelihood of future states. When state probabilities can legitimately be prescribed, it is a decision under risk, and the expected-value criterion is appropriate. When probabilities are unattainable, other criteria such as minimax, maximin, and minimax-regret must be applied to select the preferred alternative [1]. As indicated in the payoff matrix of Table 1, no probabilities are available for states of nature $N_1, N_2, \ldots, N_n$ for the decision condition of uncertainty. Each state is assumed to be independent of other states and immune to manipulation by the decision maker. Because there are no probabilities associated with the states of nature, selection of an alternative depends on the evaluation strategy employed by the decision maker.

Table 1. The Payoff Matrix

Strategy	State of Nature			
	N_1	N_2	...	N_n
S_1	O_{11}	O_{12}	...	O_{1n}
S_2	O_{21}	O_{22}	...	O_{2n}
. .	. .	. .	. .	. .
S_n	O_{n1}	O_{n2}	...	O_{nn}

After the payoff matrix has been developed, the next step is to check for dominance. If one alternative produces a greater payoff than another alternative for every state of nature, then a rational decision maker would never select the lower-paying course of action. The higher-paying alternative is said to dominate the lower one. Every researcher in operations research (OR) models knows that data in this field of human science are deterministic or random or uncertain. Of course, if measurements are available, the scientist must use such strong data but, in many case, a lot of data are weaker and subjectivity is necessary. Fuzzy sets-and specially, fuzzy numbers-is a good tool for the OR analyst facing partial uncertainty and subjectivity [2].

2. Fuzzy Sets and Decision Criteria Under Uncertainty

Fuzzy logic is a branch of mathematics that incorporates uncertainty into decision variables. In traditional mathematics, variables have exact, or "crisp" values associated with them. In fuzzy logic, variables do not have precise values but instead associate a variable with a grade of membership in a set of values. The function that represents the relationship between a value and its grade of membership in a particular set is called a membership function. The sets used in fuzzy logic are called fuzzy sets because the values of the variable they represent are no longer "crisp" values but instead are represented by a membership function relating various possible values to a grade of membership.To deal with the vagueness of human thought, Zadeh [3] first introduced the fuzzy set theory. A major contribution of fuzzy set theory is its capability of representing vague knowledge. The theory also allows mathematical operators and programming to apply to the fuzzy domain.

A fuzzy number is a normal and convex fuzzy set with membership function $\mu(x)$ which both satisfies

normality: $\mu(x)=1$, for at least one $x \varepsilon R$

and

convexity: $\mu(x') \geq \mu(x_1) \wedge \mu(x_2)$

where $\mu(x) \varepsilon [0,1]$ and $\forall x' \varepsilon [x_1, x_2]$.

Triangular fuzzy number (TFN) is a special type of fuzzy number. A TFN is designed as: P=(a,b,c). It is graphically depicted in Figure 1. The parameters a,b, and c respectively denote the smallest possible value, the most promising value, and the largest possible value that describe a fuzzy event. The extended algebraic operations of TFNs can be found in [4], [5], [6], [7].

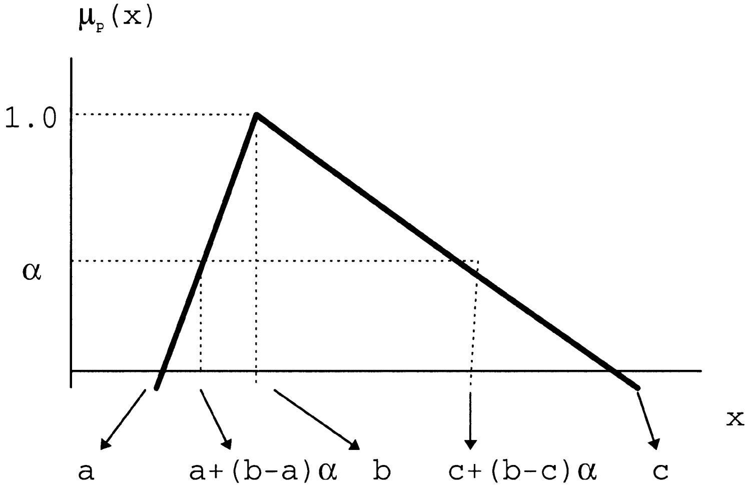


Figure 1. A Triangular Fuzzy Number

Each TFN has linear representations on its left and right side such that its membership function can be defined as:

$$\mu(x) = \begin{cases} 0 & , \quad x<a \\ (x-a)/(b-a) & , \quad a\leq x\leq b \\ (c-x)/(c-b) & , \quad b\leq x\leq c \\ 0 & , \quad x>c \end{cases} \qquad (1)$$

There are a number of methods that are devised to rank TFNs including Chang's [8] method, Kaufmann and Gupta's [9] method, Jain's [10] method, and Dubois and Prade's method [11]. Kaufmann and Gupta's [9] method and Jain's [10] method can be summarized as in the following: Kaufmann and Gupta [9] suggest three criteria for ranking TFNs with parameters (a,b,c). The dominance sequence is determined according to priority of: 1) Comparing the ordinary number:(a+2b+c)/4, 2) Comparing the mode, (the corresponding most promising value), b, of each TFN, 3) Comparing the range, c-a, of each TFN. The preference of TFNs is determined by the amount of their ordinary numbers. The TFN with the larger ordinary number is preferred. If the ordinary numbers are equal, the TFN with the larger corresponding most promising value is preferred. If TFNs have the same ordinary number and most promising value, the TFN with the larger range is preferred. Jain [10] proposes a dominance rule using maximizing set, M, of n TFNs. The maximizing set, M, is a linear representation which connects the point from the largest possible value, r_{max}, in the set with degree of membership equal 1, to the point of smallest possible value in the set with degree of membership equal 0. The preference of membership is determined by the height of the intersections of M and the right representation of each project. This method is graphically depicted in Figure 2 where $A_1>A_2>A_3$ [12]. In the following sections, first, decision criteria under uncertainty will be explained briefly, and then triangular fuzzy numbers will be used instead of crisp numbers because TFNs are very convenient and easily implemented. An example for every criterion will be given. There isn't any paper published on this subject in literature.

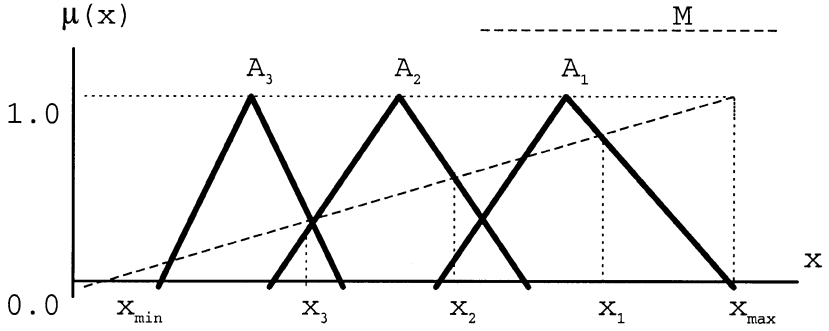


Figure 2. Jain's Method of Comparing Three TFNs

3. Fuzzy Minimax and Maximin Criteria

A minimax decision minimizes the maximum loss. This pessimistic philosophy dictates that attention be focused only on the most damaging outcomes in order to limit the damage as much as possible. The maximin rule suggests that the decison-maker determine the minimum payoff associated with each alternative and then select the alternative which maximizes the minimum profit. The maximin rule is also conservative and pessimistic, for it directs attention to the worst outcome as desirable as possible [13]. In the case of classical minimax or maximin rule, a single outcome is assigned for each alternative with respect to each state of natures. These outcomes are the possible values estimated by experts. It is clear that using a sequence of intervals of confidence instead of a sequence of ordinary numbers will give more reliable results. Therefore, in the case of fuzziness, triangular fuzzy numbers are very appropriate to use instead of those single possible outcomes. The following example shows how fuzzy numbers can be used for minimax and maximin rules.

Example 1. Consider the following payoff matrix. Select the best alternative with regard to fuzzy minimax and fuzzy maximin rule.

Table 2. The Payoff Matrix of Example 1

Alternative	State of Nature			
	N1	N2	N3	N4
S1	(2, 4, 8)	(-2,-1,0)	(-1,1,3)	(-1, 1, 2)
S2	(2, 4, 7)	(-2, 0,1)	(-3,0,1)	(3, 6, 7)
S3	(4, 5, 6)	(-4, 0,2)	(-1,0,1)	(0, 2, 5)

First, let us determine which alternative would maximize the minimum possible payoff. The minimum possible payoff for alternative S1 is (-2, -1, 0), for alternative S2 is (-3, 0, 1), for alternative S3 is (-4, 0, 2). Hence, alternative S3 will be chosen as maximizing these minimum payoffs. Kaufmann and Gupta's [9] ranking method has been used while ranking the triangular fuzzy numbers in Table 2. The ordinary number of (-2, -1, 0) is (-2+2(-1)+0)/4=-1, for (-3,0,1) is (-3+2.0+1)/4=-0.5, for (-4,0,2) is (-4+2.0+2)/4=-0.5. The second step is to rank the fuzzy numbers according to the most promising value. The most promising values of (-3,0,1) and (-4,0,2) are also equal to each other. The third step is to rank the fuzzy numbers according to the range (c-a). The range of (-3,0,1) is 1-(-3)=4, for (-4,0,2) is 2-(-4)=6. Thus, the alternative S3 is chosen. Second, we will determine which alternative would minimize the maximum possible payoff. The maximum possible payoff for alternative S1 is (2,4,8), for alternative S2 is (3,6,7), and for alternative S3 is (4,5,6). If Kaufmann and Gupta's [9] ranking method is again used, the ordinary number of (2,4,8) is calculated as (2+2(4)+8)/4=4.5, for (3,6,7) is calculated as (3+2(6)+7)/4=5.5, and for (4,5,6) is calculated as (4+2(5)+6)/4=5. Hence, alternative S1 will be chosen as minimizing these maximum payoffs.

4. Fuzzy Maximax and Minimin Criteria

A maximax philosophy is one of optimism and adventure. According to the maximax rule, the decision maker first selects the maximum payoff (profit) that is related to each alternative and then chooses the alternative that maximizes the maximum payoff. A minimum philosophy is also based on an extremely optimistic (or nonconservative) view of the outcome of nature. The minimin rule is similarly used in the case of costs (or losses). Accordingly, the decision maker first selects the minimum cost that is related to each alternative and then chooses the alternative that minimizes the minimum cost [14]. Triangular fuzzy numbers can be used instead of single values of outcomes, as mentioned before. Example 2 shows how fuzzy numbers can be used for minimin and maximax rules.

Example 2. Consider the payoff matrix in Example 1. Select the best alternative using fuzzy minimin and maximax criteria. Use Kaufmann and Gupta's [9] ranking method. The minimum possible payoff for alternative S1 is (-2,-1,0), for alternative S2 is (-3,0,1), for alternative S3 is (-4,0,2). The ordinary numbers of (-2,-1,0), (-3,0,1), and (-4,0,2) are -1, -0.5, and -0.5 respectively. Hence, alternative S1 will be chosen as minimizing these minimum payoffs. The maximum possible payoff for alternative S1 is (2,4,8), for alternative S2 is (3,6,7), and for alternative S3 is (4,5,6). The ordinary numbers of (2,4,8), (3,6,7), and (4,5,6) are 4.5, 5.5, and 5 respectively. Hence, alternative S2 will be chosen as maximizing these maximum payoffs.

5. Fuzzy Hurwicz Criterion

The Hurwicz criterion can be used for those situations in which the decision maker feels neither extremely pessimistic. The degree of optimism is established by using a coefficient α, which is called an index of optimism, such that $0 \leq \alpha \leq 1$. The interpretation is as in Table 3.

Table 3. The Interpretation of Any Value of α

The Value of α	Decision Maker's Philosophy

α=0	Maximin Criterion
0<α<1	Compromise Opinion
α=1	Maximax Criterion

According to the Hurwicz criterion, after choosing an index of optimism α, calculate the weighted outcome (WO):

$$WO_i = (\text{Max } O_i)\ (\alpha) + (\text{Min } O_i)\ (1-\alpha) \tag{2}$$

where i shows alternatives, and O shows outcomes. Based on the results, the alternative that optimizes the weighted outcome is selected. The Hurwicz criterion can be criticized due to the difficulty of assigning a particular value to α [14]. In the case of fuzziness, triangular fuzzy numbers will be used instead of crisp numbers in the payoff matrix. Because of the difficulty of assigning a particular value to α, a triangular fuzzy number for α can be also used. In other words, the degree of optimism can be given as a membership function [15], [16]. Eq. (2) can be written as in Eq. (3) in the fuzzy case:

$$\underline{WO}_i = (\text{Max } \underline{O}_i)\ (\underline{\alpha}) \oplus (\text{Min } \underline{O}_i)\ (1-\underline{\alpha}) \tag{3}$$

Example 3. Consider the payoff matrix of Example 1. Select the best alternative using fuzzy Hurwicz criterion and Jain's [10] ranking method. The membership function for the degree of optimism is given in Eq. (4):

$$\begin{array}{lll} \mu(\alpha) = 0 & , & \alpha \leq 0.6 \\ \quad = 10\alpha - 6 & , & 0.6 \leq \alpha \leq 0.7 \\ \quad = 8 - 10\alpha & , & 0.7 \leq \alpha \leq 0.8 \\ \quad = 0 & , & \alpha \geq 0.8 \end{array} \tag{4}$$

6. Fuzzy Minimax - Regret (Savage) Criterion

The *regret* is defined as the difference between the maximum payoff that could have been received and the payoff that was actually obtained from the alternative selected. Therefore, the minimax regret rule is based on th assumption that a decision maker wants to avoid any regret or at least to minimize the maximum regret about a choice. To use the criterion, first a *regret matrix* is developed by finding out the maximum payoff for each state. Second, each payoff in the column is subtracted from the maximum payoff that has been recognized, and this process is reiterated for each column. Finally, the alternative that minimizes the maximum regret is then selected [10]. The following example shows how fuzzy minimax-regret criterion can be applied to the payoff matrix in Example 1 by using triangular fuzzy numbers.

Example 4. The regret matrix of the payoff matrix in Example 1 is given in Table 4. Select the best alternative using fuzzy minimax-regret criterion and Kaufmann and Gupta's [9] ranking method.

Table 4. Fuzzy Regret Matrix

	State of Nature				
Alternative	N_1	N_2 N	3 N_4	Maximu	m
S_1	(-4, 1, 4)	(-2, 1, 3)	(0, 0, 0)	(1, 5, 8)	(1, 5, 8)
S_2	(-3, 1, 4)	(0, 0, 0)	(-2, 1, 6)(0,	0, 0) (-2, 1, 6	)
S_3	(0, 0, 0)	(-4, 0, 5)	(-2, 1, 4)	(-2, 4, 7)	(-2, 4, 7)

The minimum value of the maximum values of states is (-2,1,6). Therefore, alternative S_2 is chosen.

7. Fuzzy Equal-Likelihood (Laplace) Criterion

When the decision maker is unwilling to assign probabilities to the states of nature, then one might assume that all possible outcomes are equally likely because there is no particular reason for one state of nature to be more, or less, likely than the others. Therefore, according to the Laplace criterion, since all possible outcomes are considered to be equally likely, one can select simply on the basis of expected outcomes as determined using equal probabilities for all outcomes [13]. Example 5 shows how fuzzy equal-likelihood criterion can be applied to the payoff matrix in Example 1.

Example 5. Consider the payoff matrix in Example 1. Select the best alternative using fuzzy equal-likelihood criterion and Jain's [10] ranking method. The fuzzy expected outcomes ($E(S_i)$, i=1,2,3) will be calculated for each alternative using equal probabilities.

$E(S_1)=(2,4,8)\ (1/4)\oplus(-2,-1,0)\ (1/4)\oplus(-1,1,3)\ (1/4)\oplus(-1,1,2)\ (1/4)$
$\quad = (-0.5, 1.25, 3.25)$
$E(S_2)=(2,4,7)\ (1/4)\oplus(-2,0,1)\ (1/4)\oplus(-3,0,1)\ (1/4)\oplus(3,6,7)\ (1/4)$
$\quad = (0, 2.5, 4)$
$E(S_3)=(4,5,6)\ (1/4)\oplus(-4,0,2)\ (1/4)\oplus(-1,0,1)\ (1/4)\oplus(0,2,5)\ (1/4)$
$\quad = (-0.25, 1.75, 3.5)$

$\mu(E(S))$

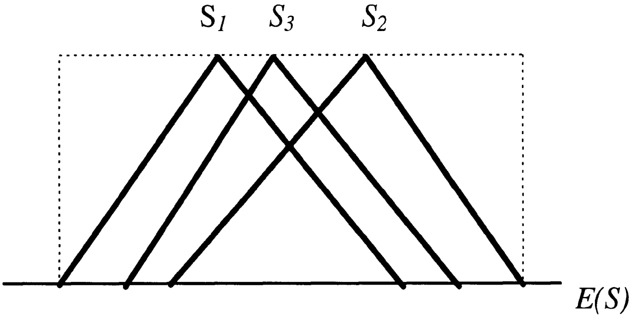


Figure 4. Jain's Ranking Method for Example 5

According to Jain's [10] ranking method, alternative S_2 is selected.

8. Conclusions

This paper presented fuzzy decision criteria under uncertainty. Fuzzy numbers have been used instead of ordinary numbers. Fuzzy numbers are a special case of fuzzy sets. Triangular fuzzy number is a special type of fuzzy number. To perform a vague predicate such as the outcome of a strategy and a state of nature, the triangular fuzzy number with three parameters is very convenient and eaisly implemented. The results obtained are more reliable than the crisp ones. A few ranking methods should be used to rank the outcomes while applying the criteria because the ranking methods in literature may give some different ranking results. Bortolan and Degani's [17] paper is a good study on ranking methods.

References

[1] Riggs, J.L., Engineering Economics, McGraw-Hill, Inc., 1982.

[2] Kaufmann, A., On the Relevance of Fuzzy Sets for Operations Research, European Journal of Operational Research, Vol.25, No.3, pp.330-335, 1986.

[3] Zadeh, L.A., Fuzzy Sets, Information and Control, Vol.8, p.338-353, 1965.

[4] Kahraman, C., Tolga, E., Ulukan, Z., Fuzzy Justification of Advanced Manufacturing Technologies, in Pre-Prints of the Nineth International Working Semimar on Productions Economics, Vol.2, pp.595-617, February 19-23, Igls/Innsbruck-Austria, 1996.

[5] Kahraman, C., Ulukan, Z, Tolga, E., Fuzzy Payback Period Analysis for the Technologies with Rapid Changes, in Proceedings of the Production and Operations Management Society and the University of Wollongong-Australia Meeting (POMS-Australia Meeting) Wollongong, Australia, July 7-10, 1996.

[6] Kahraman, C., Tolga, E., Ulukan, Z., Fuzzy Discounted Cash Flow Techniques, in Proceedings of the Fifth International Conference on Computer Integrated Manufacturing and Automation Technology (CIMAT'96), pp.65-69, May 29-31, Grenoble-France, 1996.

[7] Kahraman, C., Tolga, E., Ulukan, Z., Fuzzy Analyses in Capital Budgeting, Information Processing and Management of Uncertainty in Knowledge-Based Systems (IPMU'96), Vol. III, pp.1495-1500, Granada-Spain, July 1-5, 1996.

[8] Kaufmann, A., Fuzzy Subsets Applications in O.R. and Management, in Fuzzy Sets Theory and Applications, A. Jones, A. Kaufmann, H.-J. Zimmermann (Editors), pp.257-300, D. Reidel Publishing Company, 1986.

[9] Turksen, I.B., Wilson, I.A., A Fuzzy Set Model for Market Share and Preference Prediction, European Journal of Operational Research, Vol.82, No.1, pp.39-52, Apr. 6, 1995.

[10] Jain, R., Decision-Making in the Presence of Fuzzy Variables, IEEE Trans. Systems Man Cybernet, Vol.6, pp.698-703, 1976.

[11] Dubois, D., Prade, H., Ranking Fuzzy Numbers in the Setting of Possibility Theory, Information Sciences, Vol.30, pp.183-224, 1983.

[12] Chiu, C., Park, C.S., Fuzzy Cash Flow Analysis Using Present Worth Criterion, The Engineering Economist, Vol.39, No.2, pp.113-138, Winter 1994.

[13] Canada, J.R., White, J.A., Capital Investment Decision Analysis for Management and Engineering, Prentice-Hall, Inc., 1980.

[14] Gönen, T., Engineering Economy for Engineering Managers, John Wiley & Sons, Inc., 1990.

[15] Kahraman, c., Tolga, E., A Multi-Attribute Fuzzy Scoring Model to Select the Best Technology, in Proceedings of the Fifth International Conference on Management of Technology, pp.567-576, February 27 - March 1, Miami-Florida-USA, 1996.

[16] Kahraman, C., Ulukan, Z., Yenisey, M.M., Tolga, E., Fuzzy Economic and Strategic Design, in Proceedings of the 5th IEEE International Conference on Emerging Technologies and Factory Automation, Kauai, Hawaii, November 18-21, 1996.

[17] Bortolan, G., Degani, R., A Review of Some Methods for Ranking Fuzzy Subsets, Fuzzy Sets and Systems, Vol.15, pp.1-19, 1985.

Improving Performance of Naive Bayes Classifier by Including Hidden Variables

Bozena Stewart

Department of Computing and Information Systems
University of Western Sydney, Macarthur
Campbelltown, NSW, 2560, Australia
b.stewart@uws.edu.au

Abstract. A basic task in many applications is classification of observed instances into a predetermined number of categories or classes. Popular classifiers for probabilistic classification are Bayesian network classifiers and particularly the restricted form known as the Naive Bayes classifier. Naive Bayes performs well in many domains but suffers the limitation that its classification performance cannot improve significantly with an increasing sample size. The expressive power of Naive Bayes is inadequate to capture higher order relationships in the data. This paper presents a method for improving predictive performance of the Naive Bayes classifier by augmenting its structure with additional variables learned from the training data. The resulting classifier retains the advantages of simplicity and efficiency and achieves better predictive performance than Naive Bayes. The approach proposed here can be extended to more general Bayesian network classifiers.

1 Introduction

A central problem in machine learning, data analysis, and pattern recognition is the classification of observed instances into a predetermined number of categories or classes. Classification task requires a database of cases in which one of the attributes is a class attribute whose values are the class labels used for classification. From this database, referred to as the training data set, is constructed a classifier which is then used to predict class labels for cases whose class is unknown. The classification problem has been widely studied in statistics and Artificial Intelligence (AI) and a variety of different classification methods have been developed. The most popular approaches have been the nearest-neighbour classifiers, decision trees, neural networks, genetic algorithms and Bayesian network classifiers. This paper is concerned with a specific type of Bayesian network classifier known as the Naive Bayes classifier. The subsequent sections of the paper present a method for improving predictive performance of the Naive Bayes classifier by augmenting its structure with additional variables learned from the training data.

The general classification problem involves finding a mapping of an attribute vector to discrete values of a class attribute. The distinguishing feature of Bayesian network classifiers is that they produce probabilistic mapping. Such a mapping does not assign an instance into one particular class but computes a probability distribution giving the probability with which an instance belongs to each class. General Bayesian network classifiers are known as Bayesian networks, belief networks or causal probabilistic networks. A Bayesian network consists of two components: 1) a directed acyclic graph representing domain structure, and 2) conditional probability distributions (tables) associated with the vertices in the graph. The graph together with the conditional probability tables represent the joint probability distribution contained in the data. Using the chain rule the joint distribution can be written in the product form:

$$P(x_1, x_2, ..., x_n) = \prod_i P(x_i|\pi_{X_i}) \quad , \tag{1}$$

where lower case x_i denote values of the corresponding random variables X_i and π_{X_i} denotes an instantiation of the parents of X_i. General unrestricted Bayesian networks are unsupervised classifiers in which there is no specific class variable, all variables are treated in the same way and any one can be used as the class variable. Classification in Bayesian network classifiers involves performing probabilistic inference on the Bayesian network using one of the available probabilistic inference algorithms, for example the algorithm of Lauritzen and Spiegelhalter [7].

The Naive Bayes classifier is a special type of Bayesian network classifier, derived by assuming conditional independence of the attributes. The graphical structure of Naive Bayes is restricted so that the class attribute is the root of the tree and the directed edges lead from the root to the remaining variables (leaves). It is assumed that each attribute is conditionally independent of the remaining attributes given the class attribute. Classification in Naive Bayes determines the posterior probability distribution of the class attribute given observed values of the remaining attributes, $P(Class|A_1, A_2, ..., A_n)$.

The graph in Fig.1 represents a simple Naive Bayes classifier in which the variable C is the class variable. Using the chain rule the joint probability distribution corresponding to the graph can be written in the form:

$$P(c, a_1, a_2, a_3, a_4) = P(c)P(a_1|c)P(a_2|c)P(a_3|c)P(a_4|c) \quad . \tag{2}$$

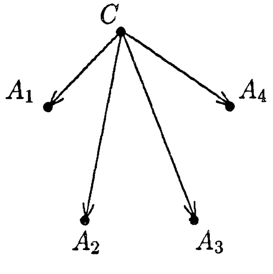


Fig.1. Naive Bayes

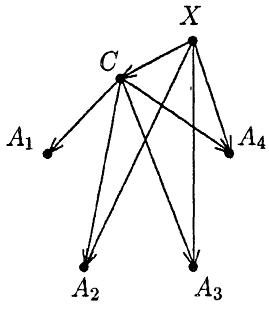


Fig.2. Naive Bayes with a hidden variable.

The Naive Bayes classifier is simple to implement, is robust with respect to noisy or missing data and performs surprisingly well in many domains. For small data sets it frequently outperforms even more sophisticated state of the art decision tree classifiers [10]. Some comparative studies have been done by Friedman and Goldszmidt [2]. One limitation of Naive Bayes is that its performance does not improve significantly with increasing sample size. The assumption of conditional independence is often not met in practice and the parameters in the model are inadequate to express all possible value combinations that occur in large data sets. Several researchers have attempted to improve the expressive power of Naive Bayes by grouping attributes into clusters (Kononenko [5], Pazzani [9]). A major weakness of these approaches has been the tendency to select large clusters which in turn leads to overfitting and incorrect predictions.

Friedman and Goldszmidt [2] developed a Tree Augmented Naive Bayes classifier (TAN) by augmenting the Naive Bayes structure with a conditional tree derived in a similar way as the Maximum Weight Spanning tree of Chow and Liu [1]. An increased number of links in the graph provides additional parameters thereby increasing the expressive power of the classifier. TAN classifier is more complex than Naive Bayes but is still highly efficient and achieves superior classification results. But the basic limitation of Naive Bayes, the inability to significantly improve predictive performance with an increasing sample size, also remains in the TAN classifier. Mutual interactions of more than three variables cannot be represented by this model and thus the higher order interactions that become apparent in large data sets are ignored.

In this paper we present a method which increases expressive power of Naive Bayes by augmenting it with "hidden" variables derived from the training data. The graph in Fig.2 illustrates such an augmented classifier. We use a generalization of Goodman's Maximum Likelihood algorithm for latent variables [3] to induce hidden variables. Experimental results indicate that this approach is practically feasible and produces good results.

The remainder of this paper is organized as follows. Section 2 outlines the problem of identifying hidden variables from data and presents our generalization of Goodman's ML algorithm, referred to as the GML algorithm, Section 3 discusses the use of hidden variables to improve predictive performance of the Naive Bayes classifier, Section 4 describes experimental results, and Section 5 discusses future work and concludes the paper.

2 Learning Hidden Variables from Data

In this section we present an algorithm for identifying hidden variables from data, derived from a procedure developed by Goodman [3] by relaxing the assumption of conditional independence.

Hidden or latent variables are variables that cannot be quantitatively measured but that influence the relationships among the observed (measurable) variables. The area of statistics concerned with the problem of identifying discrete latent variables from data is known as Latent Class Analysis. The basic assumption in the study of latent variables is that the correlations actually observed among the observed variables are due to each observed variable's relationship to the latent variable, or in terms of conditional independence, that the pairs of observed variables are conditionally independent given the latent variable. The m variable model can be represented graphically in terms of a directed tree in which the latent variable is the root and the observed variables A_i, $i = 1, ...m$ are the leaves. Equation (3) in Section 2.1 corresponds to applying the chain rule to this graphical structure.

The main algorithm used in latent class analysis to estimate latent class parameters is the maximum likelihood algorithm (ML) of Goodman [3]. In this section we present a generalization of this algorithm, referred to as the Generalized Maximum Likelihood Algorithm (GML) which uses a generalized set of Goodman's maximum likelihood equations. The generalized equations relax the assumption of conditional independence and allow the observed variables adjacent to the hidden variable to have more than one parent and hence will converge for a wider variety of probability distributions of the observed variables. The generalized equations reduce to the original equations when the variables are conditionally independent.

2.1 The Generalized Maximum Likelihood Algorithm

In order to facilitate computation, we impose the necessary condition that the parents of each variable adjacent to the hidden variable X must be adjacent only to the other variables adjacent to X and/or the latent variable itself. This model can be represented graphically by a directed acyclic graph obtained by augmenting the directed tree with additional directed edges connecting pairs of observed variables.

We consider a set of vertices $\mathbf{S} = \{A_1, A_2, ..., A_m, X\}$, where $\mathbf{S} \subseteq \mathbf{V}$, and A_1, A_2,...,A_m are observed variables which have one or more parent nodes, all of which belong to the set $\mathbf{S}$, and X is a latent variable which has no parent nodes. We denote the set of parents of node A_i by Π_{A_i}, and their instantiation by π_{A_i}. If all A_i have only the hidden variable as the parent, the equations reduce to the original equations of Goodman [3]. We obtain the following generalized equations:

$$\hat{P}(a_1, a_2, ..., a_m, x) = \hat{P}(x)\hat{P}(a_1|\pi_{A_1})\hat{P}(a_2|\pi_{A_2})...\hat{P}(a_m|\pi_{A_m}) \quad , \tag{3}$$

$$\hat{P}(a_1, a_2, ..., a_m) = \sum_{\mathbf{x}} \hat{P}(a_1, a_2, ..., a_m, x) \quad , \tag{4}$$

$$\hat{P}(x|a_1, a_2, ..., a_m) = \frac{\hat{P}(a_1, a_2, ..., a_m, x)}{\hat{P}(a_1, a_2, ..., a_m)} \quad , \tag{5}$$

$$\hat{P}(x) = \sum_{\mathbf{a_1, a_2, ..., a_m}} p(a_1, a_2, ..., a_m)\hat{P}(x|a_1, a_2, ..., a_m) \quad , \tag{6}$$

$$\hat{P}(\pi_{A_i}) = \sum_{\substack{\mathbf{x},\mathbf{a_j} \\ X,A_j \notin \Pi_{A_i}}} p(a_1, a_2, ..., a_m)\hat{P}(x|a_1, a_2, ..., a_m) \quad , \tag{7}$$

$$\hat{P}(a_i|\pi_{A_i}) = \frac{\sum_{\substack{\mathbf{x},\mathbf{a_j} \\ X,A_j \notin \{A_i \cup \Pi_{A_i}\}}} p(a_1, a_2, ..., a_m)\hat{P}(x|a_1, a_2, ..., a_m)}{\hat{P}(\pi_{A_i})} \quad , \tag{8}$$

for $i = 1, 2, ..., m$.

Upper case letters denote random variables, A_1, A_2, ..., A_m represent the observed variables and the letter X an unknown hidden variable. Lower case letters in the probability distributions denote individual values of the corresponding random variables. The bold lower case letters in the summation formulae indicate that the sum is computed for all possible value combinations of the corresponding variables. The notation used in the summation formula in (7) implies that the sum is evaluated for all combinations of values of the variables X, A_j, $j = 1, ..., m$, where X and A_j are not members of the parent set of the variable A_i. Similarly, in (8) the sum is computed for all combinations of values of X, A_j, $j = 1, ..., m$, where X and A_j are not members of the set containing the variable A_i and all its parents Π_{A_i}.

Circumflexes denote the maximum likelihood estimates of the corresponding parameters in the latent class model. We have used a lower case p to denote the joint probability distribution of the observed variables $A_1, A_2, ..., A_m$, to distinguish it clearly from the estimated distributions. This distribution is computed from the data.

The Generalized Maximum Likelihood algorithm is summarized in the following pseudocode.

```
procedure GML
  begin
    repeat
       1. Generate a starting solution vector of the form

          $\hat{P} = \{\hat{P}(x), \hat{P}(a_1|\pi_{A_1}), \hat{P}(a_2|\pi_{A_2}), ..., \hat{P}(a_m|\pi_{A_m})\}$ .

       2. Repeatedly evaluate equations (3) through (8)
          until convergence or a specified number of
          iterations has been reached.
    until the required number of trials
  end
```

For the case of conditionally independent attributes Goodman proved that when the components of the starting solution lie in the interval $(0, 1)$, the iterative procedure will converge and yield a solution to the maximum likelihood equations. The resulting solution will be either the maximum likelihood estimate, or some other solution to this system of equations. The main difficulty with this method is that it does not always give a unique solution, but sometimes produces different solutions depending on the starting trial values. That is, the computed maximum likelihood estimate may represent a local, rather than the global maximum. To overcome this problem, Goodman suggested to try various initial trial values for the vector $\hat{P}$, and for each solution compute the χ^2 statistic based on the likelihood ratio and take as the maximum likelihood solution the solution that minimizes the χ^2 statistic. The likelihood ratio χ^2 statistic has the form

$$\chi^2 = 2 \sum_{\mathbf{a_1,a_2,\ldots,a_m}} f(a_1,a_2,...,a_m)log(f(a_1,a_2,...,a_m)/\hat{F}(a_1,a_2,...,a_m)) \quad , \quad (9)$$

where $f(a_1,a_2,...,a_m) = np(a_1,a_2,...,a_m)$, $\hat{F}(a_1,a_2,...,a_m) = n\hat{P}(a_1,a_2,...,a_m)$ and n is the number of examples in the database. The number of degrees of freedom is computed as the difference of the number of parameters in the unrestricted model and the number of parameters in the estimated model.

3 Using Hidden Variables to Improve Performance of the Naive Bayes Classifier

We now show how the classification performance of the Naive Bayes classifier can be improved by augmenting its structure with hidden variables. As we mentioned previously, one limitation of Naive Bayes is its inadequate expressive power caused by a relatively small number of parameters in the model. Although a small number of parameters results in efficient probabilistic inference, it has the undesirable effect that the classifier cannot significantly improve its performance as more data becomes available.

The basic idea of our approach is to include in the graph of Naive Bayes additional variables which embody higher order relationships among several variables in the graph. The new variables are induced from the training data set using the GML algorithm described in Section 2.1. Essentially, we use the GML algorithm to fit a hidden variable to a subset of variables in the graph so that the joint distribution of the variables in the subset computed using the hidden variable is as close as possible to the observed joint distribution of the subset variables.

In this paper we propose an algorithm which augments Naive Bayes with one variable at a time. It can be repeatedly applied to include as many new variables as required. The first stage of the algorithm determines a subset of variables from which to induce a hidden variable. Since we are interested in improving the classification performance of the model, one of the variables should be the class variable. The subset could be selected manually by asking the user to specify the variables or automatically by optimizing some objective function. At present our implementation uses a manual selection.

The second stage of the algorithm involves learning a hidden variable from the chosen subset of variables using the GML algorithm. The graph is first augmented with a new variable and the links connecting it to each of the variables in the selected subset. The augmented graph is then directed using the Maximum

Cardinality Search algorithm (MCS) of Tarjan and Yannakakis [12] so that the hidden variable becomes the root of the resulting directed acyclic graph. For this particular graphical structure this guarantees that the necessary condition imposed on the parents of the selected variables, specified in Section 2.1, is satisfied. The joint probability distribution of the selected subset of variables corresponds to the observed joint distribution in equations (6) to (8) and is computed from the data. The user specifies the desired number of values of the hidden variable and the equations (3) through (8) are repeatedly evaluated until convergence or until some maximum number of iterations have been performed. The procedure is summarized in the following pseudocode.

```
procedure AugmentedNB
  begin
     1. Determine a subset of variables from which to
        learn a hidden variable.
     2. Augment the graph with a hidden variable and
        the associated edges connecting it to the
        variables in the selected subset.
     3. Direct the graph using MCS [12] so that
        the hidden variable becomes the root of the graph.
     4. Learn the hidden variable parameters using
        the GML algorithm.
  end
```

The addition of a hidden variable to the graph of Naive Bayes increases the maximum clique size from two to three but this does not lead to a great reduction in the efficiency of probabilistic inference, especially when the hidden variable is binary valued. To augment the resulting graph with additional hidden variables we first store the previously computed conditional distributions and then apply the GML algorithm as previously. The stored distributions are used during probabilistic inference to compute the required potential functions.

In all our experiments the algorithm converged successfully to good approximate solutions. In most cases only binary valued hidden variables and a relatively small number of iterations were required to produce satisfactory solutions.

4 Experimental Results

Table 1 summarizes the results of several experiments with real data obtained from the Machine Learning Repository at the University of California, Irvine [8]. In all the experiments the hidden variable was binary valued and the subset of variables was chosen by the user at random. The maximum number of iterations was set to 20 and the number of trials to 3.

In the experiments in Table 1 the inclusion of a single hidden variable resulted in an improvement in the predictive performance of the classifier. The magnitude of improvement is dependent on the choice of the subset from which the hidden

Table 1. Experiments with Naive Bayes classifier augmented with one hidden variable.

Data Set Name	Number of Cases Training	Testing	Accuracy on Test Set (%) Naive Bayes	Augmented NB	Variables in Subset
soybean	307	376	73.9	89.6	(1,13,17,20,30,34)
letter	16000	4000	74.0	75.4	(1,8,10,15)
segment	2100	210	94.7	96.7	(1,3,14,17)
landsat	4435	2000	81.6	82.5	(10,15,20,30,37)
splice	2390	800	96.5	96.9	(1,31,32,33,34)
iris	149	149*	95.9	97.3	(1,2,3,4,5)
crx	490	490*	88.4	89.4	(3,5,8,12,16)

Note: * denotes that the same data set was used for training and testing.

variable is induced. In the above experiments the choice was made arbitrarily by the user and so the results are not optimal. In some cases the accuracy can be increased by using a hidden variable with three or more values. We would expect that adding more hidden variables would lead to further improvements.

5 Conclusion and Future Work

This paper presented a new approach to improving expressive power of the Naive Bayes classifier by augmenting the structure of Naive Bayes with additional variables induced from the training data. The new variables are learned using a generalization of the maximum likelihood algorithm of Goodman [3]. Experimental results have shown that this method leads to improved classification results.

Our future work will include extending the method so that it can be used to augment Naive Bayes with more than one hidden variable. We will also develop a technique for selecting the subsets of variables for learning hidden variables automatically by the program.

Hidden variables can be used to augment graphical structure of the TAN classifier and general Bayesian network classifiers. The basic approach is similar, except that some further restrictions on the choice of the subset of variables are required. Our future work will involve applying this method to the TAN classifier and more complex Bayesian networks.

References

1. Chow, C.K., Liu, C.N.: Approximating discrete probability distributions with dependence trees. IEEE Trans. Inf. Theory **14** (1968) 462–467
2. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. To appear in Machine Learning.
3. Goodman, L.A.: Exploratory latent structure analysis using both identifiable and unidentifiable models. Biometrica **61** (1974) 215–231
4. Heckerman, D., Geiger, D., Chickering, D.M.: Learning Bayesian networks: The combination of knowledge and statistical data. Machine Learning **20** (1995) 197–243

5. Kononenko, I.: Semi-naive Bayesian classifier. In: Proceedings of the Sixth European Working Session on Learning. Springer-Verlag, Berlin, 1991, pp.206–219
6. Lam, W., Bacchus, F.: Using causal information and local measures to learn Bayesian networks. In: Heckerman, D., Mamdani, A. (eds.): Uncertainty in Artificial Intelligence. Proceedings of the Ninth Conference, Morgan Kaufmann, San Mateo, California, 1993, pp.243–250
7. Lauritzen, S.L., Spiegelhalter, D.J.: Local computations with probabilities on graphical structures and their applications to expert systems (with discussion). J. Roy. Statist. Soc. Ser. B **50** (1988) 157–224
8. Murphy, P.M., Aha, D.W.: UCI Repository of machine learning databases. Irvine, CA: University of California, Department of Information and Computer Science, 1994
9. Pazzani, M.J.: Searching for dependencies in Bayesian classifiers. In: Fisher, D., Lenz, H. (eds.): Proceedings of the Fifth International Workshop on Artificial Intelligence and Statistics. Fort Lauderdale, FL, 1995
10. J. R. Quinlan, J.R.: C4.5:Programs for machine learning. Morgan Kaufmann, San Mateo, 1993
11. Singh. M., Valtorta, M.: An algorithm for the construction of Bayesian network structures from data. In: Uncertainty in Artificial Intelligence. Proceedings of the Ninth Conference, Morgan Kaufmann, San Mateo, California, 1993, pp.259–265
12. Tarjan, R.E., Yannakakis, M.: Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. SIAM J. Comput. **13** (1984) 566–579

A System Handling RCC-8 Queries on 2D Regions Representable in the Closure Algebra of Half-Planes[*]

Brandon Bennett, Amar Isli and Anthony G. Cohn

Division of AI, School of Computer Studies
University of Leeds, Leeds LS2 9JT, UK
e-mail: {brandon,isli,agc}@scs.leeds.ac.uk

Abstract. The paper describes an algebraic framework for representing and reasoning about 2D spatial regions. The formalism is based on a *Closure Algebra* (CA) of half-planes — i.e., a Boolean Algebra augmented with a *closure* operator. The CA provides a flexible representation for polygonal regions and for expressing topological constraints among such regions. The paper relates these constraints to relations defined in the 1st-order Region Connection Calculus (RCC). This theory allows the definition of a set of eight topological relations (RCC-8) which forms a partition of all possible relations between two regions. We describe an implemented algorithm for determining which of the RCC-8 relations holds between any two regions representable in the CA. One application of such a system is in Geographical Information Systems (GIS), where often the data is represented quantitatively, but it would be desirable for queries to be expressed qualitatively in a high level language such as that of the RCC theory.

Keywords: Closure Algebra, Half-Plane, RCC, GIS.

1 Introduction

There are a number of reasons why qualitative information may be useful in a spatial information system. One is that sometimes only partial information may be available. For example, we may know that one region is *disconnected* from another without knowing the precise geometry of both regions. Although it may be possible to capture this kind of information in terms of combinations of equations and inequalities between polynomials of numerical coordinates, the resulting representation would be highly complex and moreover would be dependent on the particular coordinate system and data structures employed. A qualitative representation allows information to be stored directly in terms of 'facts', so the fact that regions a and b are disconnected might be stored simply as '$\mathsf{DC}(a, b)$'. Even if all the spatial data stored by a system is in the form

[*] This work was supported by the EPSRC under grant GR/K65041.

of precise geometrical coordinates, mechanisms for extracting and manipulating qualitative relationships between data objects are necessary for many tasks that a system might be expected to perform (Bennett, Cohn and Isli 1997). The most obvious function of this kind is to answer queries containing qualitative spatial relations ('Find all forests that are *next to* a marsh'). The reasoner to be described deals with quantitative data stored as polygons, which may correspond to a boundary map, a political map or any combination of such maps; its functionalities consist of extracting qualitative information from the quantitative data, such as finding all forests next to a marsh, checking whether a region is a subregion of another, or computing what might be called the qualitative database of a map (the RCC-8 relation between every two regions in the map).

The data in Geographical Information Systems (GIS) is generally represented quantitatively, either in vector or raster format (we assume the former in the paper). It may consist, for instance, of England's or Europe's geographical map. Given such data we may wish to answer queries such as:

1. Is Leeds in West Yorkshire?
2. Does Brussels lie on the Belgian boundary with France?
3. What are the neighbouring countries of the European Community?

We represent regions as unions of convex polygons and these polygons are in turn represented as intersections of half-planes. Real-world regions may have curved or irregular boundaries which cannot be exactly represented in this way; however, such regions can be approximated up to any desired accuracy by polygonal regions. These regions can be represented algebraically by terms in the Closure Algebra (CA) of half-planes. A CA (McKinsey and Tarski 1944) consists of a Boolean Algebra (BA) augmented with a closure operator *cl* satisfying certain constraints. Topological relationships between regions can be specified by means of equations between terms in the CA. Hence, whether a pair of regions stand in a particular topological relationship can be determined by testing whether a corresponding CA equation holds. We give a procedure for testing such equations by reducing terms to a normal form.

The paper is organised as follows. Section 2 provides a brief background on the RCC theory and the RCC-8 relations set (Cohn, Bennett, Gooday and Gotts 1997a, Randell, Cui and Cohn 1992). Section 3 is about qualitative representations and quantitative data in spatial information systems. Section 4 defines the CA of half-planes. Our query answering procedure is then described in section 5. Section 6 illustrates our implementation in action. Finally, section 7 is the conclusion.

2 The RCC Theory and the RCC-8 Relations

The Region Connection Calculus (RCC) (Randell et al. 1992, Cohn, Bennett, Gooday and Gotts 1997b, Cohn et al. 1997a) is a 1st-order theory of spatial regions based on the primitive relation of connectedness $\mathsf{C}(x, y)$ (read 'region x is connected to region y'), which is axiomatised to be reflexive and symmetrical.

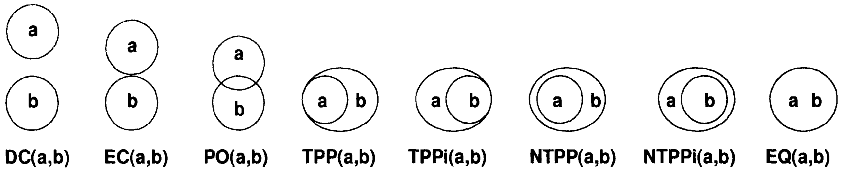


Fig. 1. The RCC-8 basic relations.

The theory enables a large class of topological relationships to be defined. In particular one can define: Dis-Connection (DC), External Connection (EC), Partial Overlap (PO), Tangential Proper Part (TPP), Non-Tangential Proper Part (NTPP) and Equality (EQ). Of these, the proper part relations, being asymmetric, have inverses denoted *Ri*. This gives a set of eight relations known as RCC-8. The intended meanings of each of these are illustrated in Figure 1. The same relation set[2] has independently been identified as significant in the context of Geographical Information Systems (Egenhofer and Franzosa 1991, Egenhofer 1991, Clementini, Sharma and Egenhofer 1994).

Any pair of regions is related by exactly one of the relations in RCC-8 (we say that such a relation set is *Jointly Exhaustive and Pairwise Disjoint* (JEPD)). RCC-8 relations can be seen as the spatial counterpart of Allen's thirteen relations which can hold between temporal intervals (Allen 1983). Certain disjunctions of these relations are also useful in describing spatial configurations in more general terms: for instance two regions are discrete ($\mathsf{DR}(x, y)$) if they are either disconnected or externally connected; one region is a proper part of another ($\mathsf{PP}(x, y)$) if it is either a tangential or a non-tangential proper part; a region is part ($\mathsf{P}(x, y)$) of another if it is either a proper part or the two regions are equal.

3 Qualitative Representations and Quantitative Data

A purely qualitative spatial information system would be based on a qualitative representation of spatial and other properties of objects and would provide an inference mechanism for determining whether any given qualitative fact follows from some set of such facts. This system could be used to answer queries relative to a qualitative database.[3] However, a qualitative reasoning system need not be employed in isolation from coordinate-based geometrical information and other kinds of numerical data. Indeed, for many useful functions, numerical information is essential. Moreover, as explained in (Bennett et al. 1997), the combination of qualitative and quantitative representations promises to be a powerful tool in system design. It also enables useful and novel program functionality based on

[2] Actually Dornheim (1996) has distinguished slight differences in their possible interpretations.

[3] Examples of systems for computing inferences among topological relations can be found in (Bennett 1994) and (Bennett 1996).

two possible interfacing procedures: one in which the truth of a qualitative fact can be tested with respect to a quantitative database; and the other in which a set of qualitative facts is generated from a quantitative database.

In current computer systems, representation of spatial information is based almost entirely on numerical coordinates and parameters. Nevertheless, a high proportion of tests made on this data are actually used to test qualitative relations between data objects. For example, we may wish to test whether two line segments cross. This is a qualitative relationship between the segments. To determine whether a qualitative relationship such as this holds between entities an algorithm is needed which will operate on numerical data-structures so as to extract the required information. In many cases — including the case of the crossing line segments – this can be achieved by formulating the relationship in terms of a Boolean combination of equalities and inequalities involving the coordinates of points; in other cases more complex iterative routines will be required.

Whilst it is possible, on a case-by-case basis, to devise algorithms to extract specific qualitative information, when needed, from quantitative data-structures, it would be far preferable to have a general purpose method of testing all qualitative relationships which one may encounter. A qualitative representation, whose interpretation is linked in a precise way to the content of quantitative data-structures can go some way towards providing this capability. This paper aims to provide such a representation by employing the formalism of Closure Algebra as an interface between the qualitative relationships expressible in the RCC language and a quantitative database of polygonal regions.

4 The Closure Algebra of Half-Planes

A Closure Algebra is a structure, $\langle \mathcal{U}, \emptyset, \cup, \cap, ^{-}, cl \rangle$, where $\langle \mathcal{U}, \emptyset, \cup, \cap, ^{-} \rangle$ is a Boolean algebra and cl is a closure operator obeying the following equations:

$$\begin{aligned} X \cup cl(X) &= cl(X) \\ cl(cl(X)) &= cl(X) \\ cl(\emptyset) &= \emptyset \\ cl(X \cup Y) &= cl(X) \cup cl(Y) \end{aligned}$$

The elements of a closure algebra can be interpreted as subsets of an arbitrary topological space. A CA equation which follows from the cl axioms and the axioms of Boolean Algebra must hold in every topological space. Our intended interpretation is the 2D Euclidean plane, $\mathbb{R}^2$. Elements of the algebra can be interpreted as sets of points, and these can be identified with spatial regions. The unit element, $\mathcal{U}$, is then the universe of all points. Under this interpretation the symbols '$\cup$', '$\cap$' and '$^{-}$' have their usual meaning as operators on sets. $\overline{X}$ is the complement of X w.r.t. $\mathcal{U}$. An interior operator int can be defined by $int(X) \equiv_{\text{def}} \overline{cl(\overline{X})}$ and a subtraction operation by $T_1 - T_2 \equiv_{\text{def}} T_1 \cap \overline{T_2}$.

Any set of subsets of the plane, which is closed under the CA operations, forms the domain of a CA. The largest such algebra consists of arbitrary subsets of the plane but, since we are interested only in polygonal regions, we can work with a smaller sub-algebra. We thus consider the concrete CA generated from the set of all (closed[4]) half-planes in $\mathbb{R}^2$. Every polygonal region may be identified with some Boolean combination of half-planes. Amongst the elements of the half-plane CA are all the points (i.e. singleton point-sets) of $\mathbb{R}^2$, since each point is the intersection of three closed half-planes. Points are the atomic elements of the algebra.

To use this algebraic framework to describe polygonal regions we employ expressions which refer directly to half-plane elements of the algebra. A half-plane is represented by a term $hp([P_1, P_2, closed\_or\_open])$, where $P_1 \neq P_2$. This denotes the left half-plane delimited by the (directed) straight line $[P_1, P_2]$ incident with points P_1 and P_2 (a point is represented as the list $[X, Y]$ of its coordinates). *closed_or_open* is a flag indicating whether the half-plane is closed or open. To get the complementary half-plane, it is sufficient to swap the points P_1 and P_2, and replace *closed* by *open* or *vice versa.* A closed half-plane includes the points on its delimiting line; an open half-plane does not.

In many real applications involving spatial information, polygons are stored as lists of their vertex points. Hence, it is convenient to augment our half-plane CA with terms of the form $poly([P_1, \ldots, P_n])$ corresponding directly with this data-type. When terms are converted to normal form (e.g. in order to test equations) these terms are translated into unions of intersections of closed half-planes. To do this, the polygon is first split into convex parts; the conversion of a convex vertex list to an intersection of half-planes is then trivial.

The main computational benefit of using the CA generated by half-planes rather than the algebra of arbitrary subsets of $\mathbb{R}^2$ is that it allows for easy manipulation of the closure operator, *cl*. In fact, as we shall see in the next section, given that in our representation both open and closed half-planes are basic constituents of CA terms, the *cl* operator can be eliminated from any term. Further results on the use of CA for representing topological constraints can be found in (Bennett 1998).

5 The Query Answering System

The CA formalism enables many RCC relations to be represented as equations and inequations between CA terms. Equations corresponding to RCC-8 relations can be specified as in Table 1.

The system evaluates queries by testing the truth of a series of equations between CA terms. This problem can immediately be reduced to that of determining whether a term denotes the empty region $\emptyset$ because of the equivalence:

$$\alpha = \beta \qquad \text{if and only if} \qquad (\alpha \cap \overline{\beta}) \cup (\beta \cap \overline{\alpha}) = \emptyset \, .$$

We now give a procedure for testing the emptiness of any term in the half-plane CA. First we consider the case of purely Boolean terms, which do not contain the

[4] Since a CA is closed under closure and interior operations, the same algebra is generated from either the closed or the open half-planes.

RCC Rel.	Corresponding CA equations and inequations
$\mathsf{DC}(x, y)$	$cl(T_1) \cap cl(T_2) = \emptyset$
$\mathsf{EC}(x, y)$	$int(T_1) \cap int(T_2) = \emptyset \wedge cl(T_1) \cap cl(T_2) \neq \emptyset$
$\mathsf{PO}(x, y)$	$cl(T_1) - cl(T_2) \neq \emptyset \wedge cl(T_2) - cl(T_1) \neq \emptyset \wedge int(cl(T_1) \cap cl(T_2)) \neq \emptyset$
$\mathsf{TPP}(x, y)$	$cl(T_1) - cl(T_2) = \emptyset \wedge cl(T_2) - cl(T_1) \neq \emptyset \wedge cl(T_1) \cap cl(\overline{T_2}) \neq \emptyset$
$\mathsf{TPPi}(x, y)$	$cl(T_2) - cl(T_1) = \emptyset \wedge cl(T_1) - cl(T_2) \neq \emptyset \wedge cl(T_2) \cap cl(\overline{T_1}) \neq \emptyset$
$\mathsf{NTPP}(x, y)$	$cl(T_1) \cap cl(\overline{T_2}) = \emptyset$
$\mathsf{NTPPi}(x, y)$	$cl(T_2) \cap cl(\overline{T_1}) = \emptyset$
$\mathsf{EQ}(x, y)$	$(T_1 = T_2)$

Table 1. The RCC-8 relations represented as CA equations and inequations; T_1 and T_2 are CA terms representing x and y, respectively.

cl operator (or the defined operator *int*). We then show how *cl* can be eliminated from any term in the algebra.

5.1 Testing Emptiness of a Boolean term

Testing the emptiness of a Boolean term plays a crucial role in our system. Given such a term, T, we transform it into a *UNF* (Union Normal Form), U, which is a union of intersections of half-planes. This *UNF* form is analogous to the DNF form of propositional logic and the transformation procedure is essentially the same as that for propositional DNF. The term T is empty if and only if every disjunct in U is empty, so the problem is transformed into how to tesi emptiness of a disjunct (i.e., an intersection of half-planes). Noting that a half-plane is a convex region of 2D space, we make use of the $n = 2$ specialisation of Helly's convexity theorem:

> **Helly's Theorem:**[5] *Let S be a set of convex regions of the n-dimensional space $\mathbb{R}^n$. If every $n + 1$ elements in S have a non-empty intersection then the intersection of all elements of S is non-empty.*

This tells us that an intersection of half-planes is non-empty, just in case every three half-planes in the intersection have a non-empty intersection. The problem thus becomes the one of testing emptiness of a conjunction of three half-planes and this is equivalent to the problem of solving a system of three linear inequalities on two variables. The problem can hence be solved using the simplex method. However, for computational reasons, we preferred to use the Fourier-Motzkin method (Fourier 1890)[6] for solving systems of linear inequalities.

The use of Helly's theorem in our reasoner confirms its importance in Qualitative Spatial Reasoning applications, which has already been shown by (Faltings 1995).

[5] Further details of Helly's theorem can be found in (Chvátal 1983).

[6] See again (Chvátal 1983, page 241) for a brief description of this method.

5.2 Eliminating the Closure Operator *cl*

Suppose we have a term $cl(T)$ where T is a purely Boolean term (i.e., T does not contain *cl* or *int*). If we transform T into UNF we get $cl(I_1 \cup \ldots \cup I_n)$, where I_i is an intersection of half-planes. Because of the axiom $cl(X \cup Y) = cl(X) \cup cl(Y)$, we can then move the *cl* operator inwards to get $cl(I_1) \cup \ldots \cup cl(I_n)$. We now need to eliminate *cl* when it operates on an intersection of half-planes. To do this, we exploit the following lemma:

> **Lemma:** *The closure of a* non-empty *intersection of half-planes is equivalent to the intersection of their closures.*

Thus *cl* can be eliminated from each $cl(I_i)$ by first testing whether I_i is empty. If $I_i = \emptyset$ then $cl(I_i) = \emptyset$, otherwise, we simply close every half-plane in I_i by replacing half-planes of the form $hp([P_1, P_2, open])$ by $hp([P_1, P_2, closed])$.

We can use this reduction to eliminate the innermost occurrences of *cl* operators from a CA term, since these operate on purely Boolean terms. Hence, by iterating this elimination, we can transform any term of the half-plane CA into a purely Boolean term of (open and closed) half-planes. The emptiness of this can then be determined by the method given in the previous section.

5.3 Removing Redundancies

For reasons of computational efficiency it is useful to reduce redundancy in terms manipulated by the system. Hence whenever we reduce a term to UNF form we make the following simplifications:

1. The disjuncts are minimal intersections of half-planes, such that removing any half-plane from any disjunct leads to a non-equivalent disjunct.
2. No disjunct in U denotes a sub-region of the region denoted by another disjunct.

Given a non-empty intersection of half-planes, a half-plane is redundant if and only if the intersection in which that half-plane is replaced by its complement is empty.

5.4 The RCC-8 Decision Tree

The primary use of our CA equation testing procedure is to determine the RCC-8 relation holding between two regions denoted by CA terms. Rather than simply testing the equations associated with each of these relations until we find the one that is true, it is more efficient to use a decision tree such as that given in Figure 2. This may allow us to find the relation by testing fewer equations. Note that the distinction between EC and PO is here made by calling an auxiliary function *dim* which returns the dimension of the region denoted by a CA term. In 2D space we have $dim(T) < 2$ iff $int(T) = \emptyset$ and $dim(T)$ can in most cases be computed more efficiently than $int(T)$.

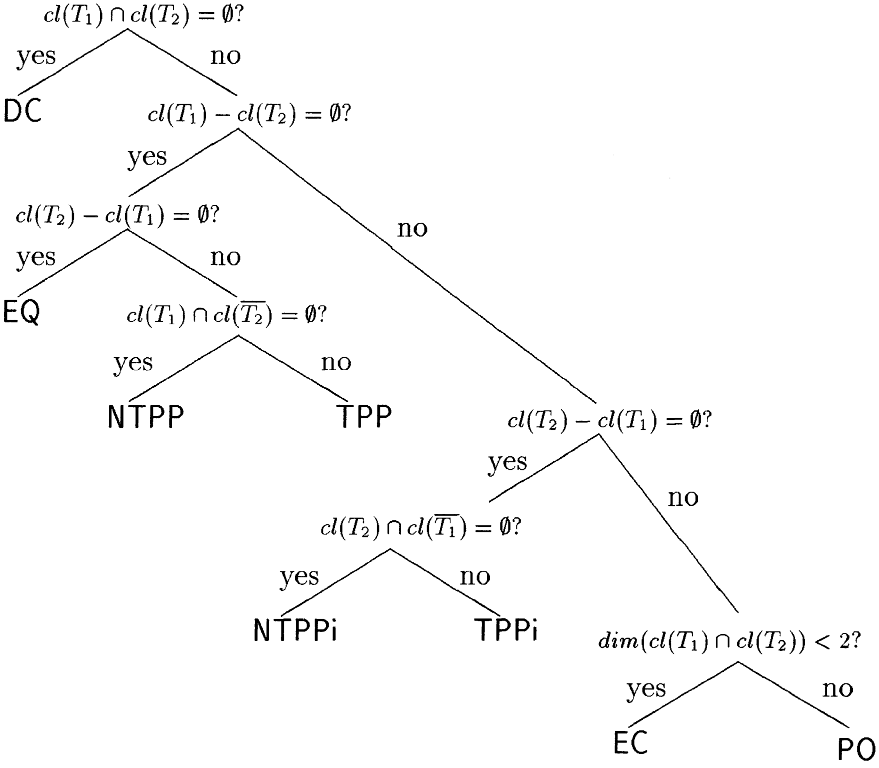


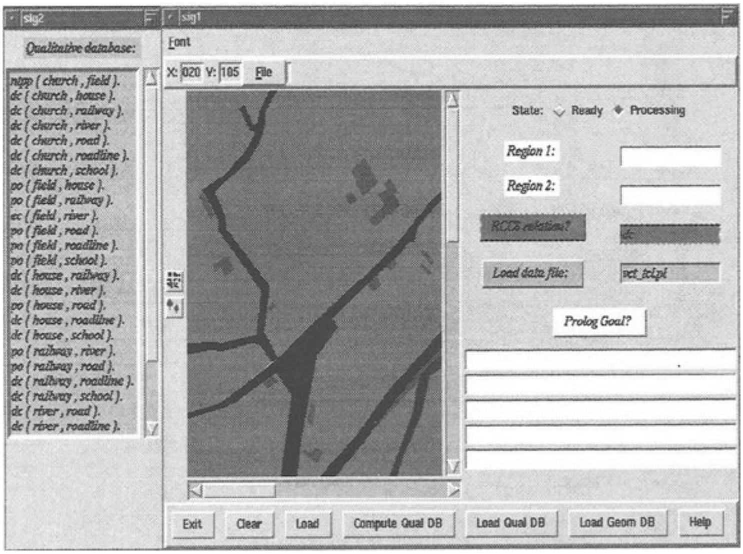
Fig. 2. Decision tree for determining the RCC-8 relation holding between regions denoted by CA terms T_1 and T_2.

5.5 Complexity of the RCC-8 Algorithm

The worst-case complexity of the query answering system is clearly exponential. Indeed, in order to check whether a Boolean term is empty, the system converts the term into a union of intersections of half-planes, and then checks whether every disjunct in the union is empty. Testing if a disjunct (i.e., an intersection of half-planes) is non empty is polynomial: it is sufficient, thanks to Helly's convexity theorem, to check whether every three half-planes in the intersection form a non empty intersection. However, the process of turning a term into *UNF* is similar to the process of turning a propositional calculus formula into DNF, which is exponential in the general case.

6 The System Implementation

Our system has been implemented in Sicstus Prolog, and makes use of the Tcl/Tk interface. Figure 3 shows some of its functionalities. For instance, clicking on the "Compute Qual DB" button computes the qualitative database (the RCC-8 relation between every two regions) of a pre-loaded data file (here 'vct_tcl.pl'), representing a geographical map. The computed qualitative database is shown on the left hand side.



Qualitative database:

```
ntpp ( church , field ).
dc ( church , house ).
dc ( church , railway ).
dc ( church , river ).
dc ( church , road ).
dc ( church , roadline ).
dc ( church , school ).
po ( field , house ).
po ( field , railway ).
ec ( field , river ).
po ( field , road ).
po ( field , roadline ).
po ( field , school ).
dc ( house , railway ).
dc ( house , river ).
po ( house , road ).
dc ( house , roadline ).
dc ( house , school ).
po ( railway , river ).
po ( railway , road ).
dc ( railway , roadline ).
dc ( railway , school ).
dc ( river , road ).
dc ( river , roadline ).
```

X: 020 Y: 185

State: ○ Ready ● Processing

Region 1:

Region 2:

RCC8 relation? dc

Load data file: vct_tcl.pl

Fig. 3. Functionalities of the system.

7 Conclusion

We have described a Closure Algebra of half-planes suitable for representing and reasoning about 2D spatial regions; as well as an implemented system determining which of the RCC-8 relations holds between two input regions representable in the CA. The system has been implemented in Sicstus Prolog, and makes use of the Tcl/Tk interface. One application of this work is in Geographical Information Systems (GIS), where the data is known, and generally given quantitatively, and the problem is often to compute the RCC-8 relation between two specified regions. We are currently working on generalising our algebraic framework so that as well as denoting concrete polygons, terms may also refer to abtitrary regions which may be constrained in their relationship to other regions although their geometry is not known.

References

Allen, J. F.: 1983, Maintaining knowledge about temporal intervals, *Communications of the ACM* **26**(11), 832–843.

Bennett, B.: 1994, Spatial reasoning with propositional logics, *in* J. Doyle, E. Sandewall and P. Torasso (eds), *Principles of Knowledge Representation and Reasoning: Proceedings of the 4th International Conference (KR94)*, Morgan Kaufmann, San Francisco, CA.

Bennett, B.: 1996, Modal logics for qualitative spatial reasoning, *Bulletin of the Interest Group in Pure and Applied Logic (IGPL)*.

Bennett, B.: 1998, Determining consistency of topological relations, *Constraints*. to appear.

Bennett, B., Cohn, A. G. and Isli, A.: 1997, Combining multiple representations in a spatial reasoning system, *Proceedings of the 9th IEEE International Conference on Tools with Artificial Intelligence (TAI-97)*, Newport Beach, CA, pp. 314–322.

Chvâtal, V.: 1983, *Linear Programming*, W.H. Freeman and Company, New York.

Clementini, E., Sharma, J. and Egenhofer, M. J.: 1994, Modeling topological spatial relations: strategies for query processing, *Computers and Graphics* **18**(6), 815–822.

Cohn, A. G., Bennett, B., Gooday, J. and Gotts, N.: 1997a, RCC: a calculus for region based qualitative spatial reasoning, *Geoinformatica*.

Cohn, A. G., Bennett, B., Gooday, J. and Gotts, N.: 1997b, Representing and reasoning with qualitative spatial relations about regions, *in* O. Stock (ed.), *Temporal and spatial reasoning*, Kluwer.

Dornheim, C.: 1996, Vergleichende analyse topologischer ansätze des qualitativen rümlichen schliësens, *Studienarbeit, Fachereich Informatik 253*, Universität Hamburg.

Egenhofer, M.: 1991, Reasoning about binary topological relations, *in* O. Gunther and H. J. Schek (eds), *Proceedings of the Second Symposium on Large Spatial Databases, SSD'91 (Zurich, Switzerland). Lecture Notes in Computer Science 525*, pp. 143–160.

Egenhofer, M. and Franzosa, R.: 1991, Point-set topological spatial relations, *International Journal of Geographical Information Systems* **5**(2), 161–174.

Faltings, B.: 1995, Qualitative spatial reaoning using algebraic topology, *in* A. U. Frank and W. Kuhn (eds), *Spatial Information Theory: a theoretical basis for GIS*, Vol. 988 of *Lecture Notes in Computer Science*, Springer-Verlag, Berlin, pp. 17–30.

Fourier, J. B. J.: 1890, Second extrait, *in* P. Gauthiers-Villars (ed.), *Oeuvres, G. Darboux*, pp. 325–328. English translation by D. A. Kohler(1973).

McKinsey, J. and Tarski, A.: 1944, The algebra of topology, *Annals of Methematics* **45**, 141–191.

Randell, D. A., Cui, Z. and Cohn, A. G.: 1992, A spatial logic based on regions and connection, *Proc. 3rd Int. Conf. on Knowledge Representation and Reasoning*, Morgan Kaufmann, San Mateo, pp. 165–176.

Cardinal Directions on Extended Objects for Qualitative Navigation

M. Teresa Escrig and Francisco Toledo

Universitat Jaume I, Campus de Penyeta Roja, 12071 Castellón (España)
e-mail: {escrigm, toledo}@inf.uji.es

Abstract. With the aim of simulating the human reasoning process about spatial aspects such as orientation, distance and cardinal directions, several qualitative models have been developed in the last years. However, there is no model for representing and reasoning with all these spatial aspects in a uniform way. In the approach presented in this paper, this integration has been accomplished thanks to consider each type of spatial relationship an instance of the Constraint Satisfaction Problem. Constraint Logic Programming instantiated to Finite Domains extended with Constraint Handling Rules is a programming paradigm which provides the suited level of abstraction to define an incremental, flexible, efficient —with polynomial cost— and general purpose Constraint Solver (CS) for each one of the spatial aspects to be integrated. Moreover, it also allows the use of point and extended objects as primitive of reasoning. The corresponding CS for cardinal directions is described in this paper. This model has been applied to the development of a Qualitative Navigator Simulator.

Keywords: Qualitative Spatial Reasoning, Constraint-Based Reasoning, Qualitative Navigation.

1 Introduction

An intelligent robot should carry out spatial reasoning for solving problems dealing with entities occupying space. If we attempt to simulate the human spatial reasoning process, it is necessary to deal with descriptions of positions of objects which contain information such as "the cinema is to your *right*, *far* from here and to the *north-west* of the city".

Many qualitative models have been developed to manage properly the imprecise knowledge about different aspects of space. For the concept of orientation, for instance, it is possible to distinguish between a set of models based on orthogonal projections ([1, 2) and non orthogonal projections [3] and a set of models not based on projections ([4, 5, 6, 7]). The set of models not based on projections are considered more cognitive due mainly to two reasons: (1) people do not reason about spatial orientation by doing projections on "inexistent" external axes. They rather think about egocentric Reference Systems (RSs) based on the asymmetry of the human body (front/back) —strongly influenced by the vision system— and both arms (right/left) [6]; and (2) in models based on orthogonal projections, it is possible to infer inconsistent knowledge (for instance, two overlapped relationships between the projections of objects in both axes, X and Y, do not imply that the two objects overlap in the two-dimensional space). The qualitative orientation models not based on projections have also been extended with the concept of distance. However, the integration of orientation, cardinal directions and distance into the same qualitative model is a problem which remains unsolved. Moreover, an important drawback of models not based on projections is that they simplify real objects to points.

Therefore our objectives are: (1) the integration of orientation, cardinal directions and distance into the same qualitative model, (2) the use of point and extended objects as primitive of reasoning, and (3) the development of a demonstrator of the application of the model to Qualitative Navigation.

The bases for the integration in the Spatial Reasoning domain has been inspired in the Temporal Reasoning field where point and interval algebra and metric information has been successfully combined using constraint satisfaction techniques [8]. These techniques have also provided the suited level of abstraction for the representation of qualitative orientation information [9] and qualitative positional information [10, 11, 12]. In this paper the representation and the reasoning process for cardinal directions will be described.

2 Bases for the integration in the Temporal Reasoning field

Many forms of temporal relationships have been integrated thanks to consider them instances of the Constraint Satisfaction Problem (CSP). The CSP can be defined such that: given a set of variables $\{X_1, \ldots, X_n\}$, a discrete and finite domain for each variable $\{D_1, \ldots, D_n\}$ and a set of constraints $\{c_{ij}(X_i, X_j)\}$, which define the relationship between every pair of variables X_i, X_j, ($1 \leq i < j \leq n$); the problem is to find an assignment of values $\langle v_1, \ldots, v_n \rangle$, $v_i \in D_i$ to variables such that all constraints are satisfied. Every different value assignment that satisfies all the constraints is called a *solution*.

Generate and test and backtracking are algorithms which solve the CSP, although in a very inefficient way. These algorithms have an exponential cost. Research in the field tries to improve efficiency of the backtracking algorithm (a review of the state of the art can be found in [13] and [14]). A set of these algorithms modify the search space before the search process starts, to make the search process easier. They are called algorithms which improve consistency. These algorithms are based on the idea of making explicit the implicit constraints by means of the constraint propagation process. Unfortunately the complete constraint propagation process is also hard, therefore the process is approximated by local constraint propagation, as path consistency. If the constraint graph is complete (that is, there is a pair of arcs, one in each direction, between every pair of nodes) it suffices to repeatedly compute paths of length two at most. This means that for each group of three nodes (i,k,j) we repeatedly compute the following operation until a fix point is reached [15]:

$$c_{ij} := c_{ij} \oplus c_{ik} \otimes c_{kj} \tag{1}$$

This operation computes the composition of constraints between nodes *ik* and *kj* and the intersection of the result with constraints between nodes *ij*. The complexity of this algorithm is $O(n^3)$, where n is the number of nodes in the constraint graph (that is, the number of objects involved in the reasoning process) [16].

Constraint Handling Rules (CHRs) are a tool which helps to write the above algorithm. They are an extension of the Constraint Logic Programming (CLP) which facilitate the definition of constraint theories and algorithms which solve them. They facilitate the prototyping, extensions, specialization and combination of CSs [15]. The part $(c_{ik} \otimes c_{kj})$ of the formula will be implemented by propagation CHRs which are of the form:

$$H_1, \ldots, H_i ==> G_1, \ldots, G_j \mid B_1, \ldots, B_k \ (i > 0, j \geq 0, k \geq 0)$$

Which means that if a set of constraints matches the head of a propagation CHR and the guards $G_1, \ldots, G_j$ are satisfied, then the set of constraints $B_1, \ldots, B_k$ is added to the initial set of constraints.

The part of the formula which referees to $(c_{ij} \oplus \ldots)$ will be implemented by simplification CHRs which are of the form:

$$H_1, \ldots, H_i <=> G_1, \ldots, G_j \mid B_1, \ldots, B_k \ (i > 0, j \geq 0, k \geq 0)$$

Which means that if a set of constraints (H') matches the head of a simplification CHR and the guards are satisfied, the set of constraints H' is substituted by the set of constraints $B_1, \ldots, B_k$. The set of constraints $B_1, \ldots, B_k$ is simpler than the set of constraints H' and preserves logical equivalence. CHRs are included as a library in ECLiPSe [17].

3 Bases for the integration in the Spatial Reasoning field

The reasoning process can be divided into two parts: the Basic Step of the Inference Process (BSIP) and the Complete Inference Process (CIP). For those models not based on projections, the BSIP can be defined in general such as: given a spatial relationship between object A with respect to (wrt) a RS, RS1, and another spatial relationship between object B wrt another RS, RS2, being object A part of the RS2, the BSIP consists of obtaining the spatial relationship of object B wrt the RS1. The RS will be different depending on the model. When more relationships among several spatial landmarks are provided, then the CIP is necessary. It consists of repeating the BSIP as many times as possible, with the initial information and the information provided by some BSIP, until no more information can be inferred.

To accomplish the integration of orientation, cardinal directions and distance into the same model we will define: (1) the representation of the spatial aspect to be integrated; (2) the BSIP for each represented spatial aspect; and (3) the CIP for this spatial aspect. In the next sections these three steps are explained for cardinal directions.

4 Cardinal directions

4.1 The representational model

The representational model for cardinal directions is based on Frank's work [7], with several differences as we will show. Cardinal direction is a binary function from two objects in space (P1, P2) that map onto a symbolic direction. The set of symbolic directions depends on the granularity. Usually humans use two levels of granularity: $d_4 = \{N, E, S, W\}$, and $d_8 = \{N, NE, E, SE, S, SW, W, NW\}$. The information represented is dir(P1,P2), where P1 will refer to the reference object and P2 to the referenced object.

For d_8 , Frank introduced the identity symbol, O, with the meaning "two points too close each other for a direction to be determined" which increases deductive power.

Besides its original meaning we are going to use d_8 with identical symbol to deal with extended objects. When the extension of an object cannot be simplified to a point, it will be delimited by the identity region which also will determine the rest of the cardinal direction regions, as it is shown in figure 1. Therefore the identity region is adapted to the size of the extended object. The prolongation of the identity region delimits the rest of the regions. In the rest of the paper we will refer always to d_8.

Given a cardinal direction for a line segment between points P1 and P2, the direction from P2 to P1 can be deduced by using the inverse operation (inv(dir(P1,P2)) = dir(P2,P1)). The inverse operation is defined in table 1. It is implemented in our approach as facts of the PROLOG database, i.e., *inv(n,s)*.

4.2 The basic step of the inference process

Given two cardinal direction relationships, between objects P1 and P2 and between objects P2 and P3, the cardinal direction relationship between objects P1 and P3 can be inferred by the BSIP (dir(P1,P2) ∞ dir (P2,P3) = dir (P1,P3)).

As example we are going to assume that P1, P2 and P3 are extended objects and dir(P1,P2)={N} (P2 is to the north of P1) and dir(P2,P3)={E} (P3 is to the east of P2) therefore we can deduce dir(P1,P3)={NE} (P3 is to the north-east of P1) as it is shown in figure 2a). However, it is possible that P2 does not occupy all the north region of P1 (as it happens in figure 2b), therefore P3 can be to the north-east of P1 but also to the north of P1, that is, dir(P1,P3)={NE,N}.

Usually the basic step of the inference process for the combination of all the cardinal directions is represented by a table. The inference table introduced in [7] only reflects the inference for the case in which all the identity regions are of the same size. Moreover, his table contains many lower case values which represent approximate reasoning, that is, "e" might mean the disjunction {NE,N} or the relation {NE}.

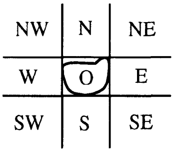


Figure 1. Cardinal directions RS for the level of granularity d_8.

Table 1. The inverse operation.

	N	NE	E	SE	S	SW	W	NW	O
inv	S	SW	W	NW	N	NE	E	SE	O

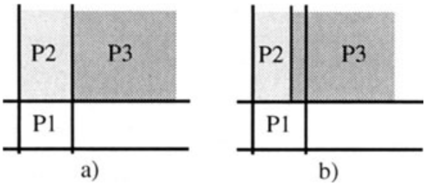


Figure 2. Example of the BSIP between dir(P1,P2)={N} and dir(P2,P3)={NE} when a) the identity regions are of the same size (dir(P1,P3)={NE}) and b) when they are not (dir(P1,P3)={N,NE}).

Table 2. The BSIP for cardinal directions. Symbol * means "all disjunctions included".

	N	NE	E	SE	S	SW	W	NW	O
N	{N}	{N,NE}	{N,NE}	{N,NE,E,SE,S,O}	{N,O,S}	{N,NW,W,SW,S,O}	{NW,N}	{NW,N}	{N}
NE	{NE}	{NE}	{NE}	{NE,E,SE}	{NE,E,SE}	*	{NE,N,NW}	{NE,N,NW}	{NE}
E	{E,NE}	{E,NE}	{E}	{E,SE}	{E,SE}	{W,SW,S,SE,E,O}	{E,O,W}	{W,NW,N,NE,E,O}	{E}
SE	{NE,E,SE}	{NE,E,SE}	{SE}	{SE}	{SE}	{SW,S,SE}	{SW,S,SE}	*	{SE}
S	{N,S,O}	*	{S,SE}	{S,SE}	{S}	{SW,S}	{SW,S}	{N,NW,W,SW,S,O}	{S}
SW	{SW,W,NW}	*	{SW,S,SE}	{SW,S,SE}	{SW}	{SW}	{SW}	{NW,W,SW}	{SW}
W	{W,NW}	{W,NW,N,NE,E,O}	{E,O,W}	{W,SW,S,SE,E,O}	{W,SW}	{W,SW}	{W}	{NW,W}	{W}
NW	{NW}	{NW,N,NE}	{NW,N,NE}	*	{NW,W,SW}	{NW,W,SW}	{NW}	{NW}	{NW}
O	{N,O}	{N,NE,E,O}	{E,O}	{S,SE,E,O}	{S,O}	{W,SW,S}	{W,O}	{W,NW,N,O}	{O}

Variable identity regions (adapted to the size of the extended objects) and exact disjunctive reasoning is represented in the inference table 2. This inference table has been implemented in our approach as facts in the PROLOG database, for instance *composition(n,ne,[n,ne])*.

4.3 The complete inference algorithm

The CIP will be the algorithm which repeatedly compute operation (1) (figure 3). i, j, k will refer now to point or/and extended spatial objects. The cardinal direction relationship dir(X,Y,L) will be implemented in our approach by the predicate ctr(X,Y,L).

Predicates ctr/3 are translated into ctr/5 by the simplification CHR (1d) where the length of the cardinal direction is added, as well as the length of the shortest path from which the constraint has been derived (length 1 means that the constraint is direct, i.e. it is user-defined). Both arguments are included to increase efficiency. The first one avoids composition between constraints which does not provide more information (simplification CHR (1e) removes the constraints which contain all the cardinal relationships). The last argument is used to restrict the propagation CHRs to involve at least one direct constraint. In (1c), *member(R,L)* nondeterministically chooses one primitive cardinal direction, R, from the disjunctive relationship, L, which implements the backtrack search part of the algorithm.

Simplification CHR (2a) performs the intersection part of the algorithm. When there exist two constraints which relate the same two spatial objects in the same order, the intersection of the corresponding relations (L1 and L2) is computed and the two initial constraints are substituted by the new one. If the constraint graph were complete, the intersection part of the algorithm would be computed with this CHR. However, normally there is no relationship between every pair of objects in the space. The lack of completeness of the graph is overcame by defining two additional

```
% Constraint definitions
(1a) constraints ctr/3,ctr/5.
(1b) label_with ctr(N,X,Y,L,I) if N>=1.
(1c) ctr(N,X,Y,L,I):-member(R,L), ctr(1,X,Y,[R],I).
(1d) ctr(X,Y,L) <=> length(L,N)|ctr(N,X,Y,L,1).
(1e) ctr(N,X,Y,L,I) <=> N=9 | true.
% Intersection
(2a) ctr(N1,X,Y,L1,I),ctr(N2,X,Y,L2,J) <=> inter(L1,L2,L3),
     length(L3,N3), min(I,J,K) | ctr(N3,X,Y,L3,K).
(2b) ctr(N1,Y,X,L1,I),ctr(N2,X,Y,L,J) <=> inv(L1,M), inter(M,L,L3),
     length(L3,N3), min(I,J,K) | ctr(N3,X,Y,L3,K).
% Composition
(3a) ctr(N1,X,Y,L1,I),ctr(N2,Y,Z,L2,J) ==> J=1, comp(L1,L2,L3),
     length(L3,M), K is I+J | ctr(M,X,Z,L3,K).
(3b) ctr(N1,Y,X,L1,I),ctr(N2,Y,Z,L2,J) ==> J=1, inv(L1,Lx),
     comp(Lx,L2,L3), length(L3,M), K is I+J | ctr(M,Y,Z,L3,K).
```

Figure 3. Algorithm to compute the CIP for cardinal directions.

simplification CHRs applying the inverse operation to the first constraint (as in (2b)) and to the second constraint.

The part $\left(c_{ik} \otimes c_{kj}\right)$ of formula (1) corresponds to the BSIP for cardinal directions explained in the previous section. It will be implemented by the propagation CHRs (3a). Again, due to the incompleteness of the constraint graph, two more propagation CHRs are defined by applying the inverse operation to the first and second constraint respectively (CHR (3b) is an example).

One important feature of this cardinal directions CS is that it can be easily modified. For instance, if we want to simulate the limited capacity of the human reasoning process to infer more than three steps in depth, we should just include in the propagation CHRs the instruction I<3, as part of the guard of the rule.

5 Using point and extended objects as reasoning primitive

Qualitative Spatial Reasoning maps spatial objects onto qualitative regions, therefore all what is known about a goal object is that it is equivalent to these qualitative regions. In the measure that these qualitative regions are bigger than a point, extended objects have to be managed. In this section a new model for representing point an extended objects as primitive of reasoning in the oriented models not based on projections is introduced. The tool used to define this model is CLP instantiated to Finite Domains, CLP(FD) [18], which is also included as library in ECLiPSe [17]. CLP(FD) allows the definition of variables with associated finite integer domains and linear terms which are integer combinations of integer domain variables. The model is based on:

- The definition of a grid which overlaps the space whose size is context dependent. It is implemented by two finite domain variables (one for each axis). This grid can be seen as a new constraint added to the system. The represented objects will be projected onto the intersections of the grid.
- Two-dimensional objects will be implemented by two finite integer domain variables (one variable for each axis corresponding to the grid).
- The straight lines included in the RSs will be represented by a set of linear terms.

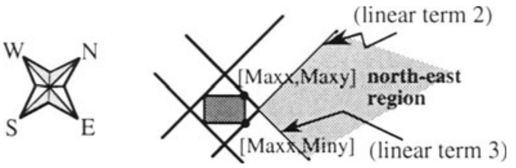


Figure 4. Linear terms which define the north-east region when Nx>Sx, Ny>Sy.

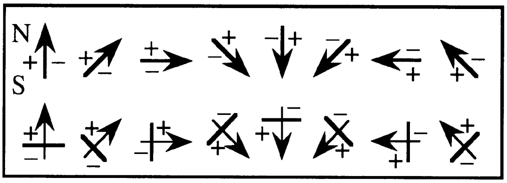


Figure 5. Study of the symbol inequalities of the linear terms.

In this paper we focus on the cardinal RS as example (see [9,11] for further examples). The cardinal RS is defined by the straight line which goes from south to north. This straight line will be determined by two points which will be implemented in our approach by two finite integer domain variables each one ([Sx,Sy] and [Nx,Ny]) whose domains contain only one value. The linear terms which represent this RS will be parallel straight lines to the one which determine the north-south by the external points of the extended objects belonging to the grid (which will be called Minx, Maxx, Miny, Maxy). We can say, for instance, that an object B is to the north-east of another reference object. Initially the domains of the variables associated to the object B (Bx and By) might contain all the values of the work space. However, by applying the following linear terms it is possible to reduce the values of the domains of the variables associated to the object B to the points belonging to the grid and included in the north-east region (figure 4):

$$(\mathrm{Bx}-\mathrm{Maxx})\times(\mathrm{Ny}-\mathrm{Sy})\#<(\mathrm{By}-\mathrm{Miny})\times(\mathrm{Nx}-\mathrm{Sx}) \tag{2}$$

$$(\mathrm{Bx}-\mathrm{Maxx})\times(\mathrm{Sx}-\mathrm{Nx})\#>(\mathrm{By}-\mathrm{Maxy})\times(\mathrm{Ny}-\mathrm{Sy}) \tag{3}$$

The cardinal RS can be oriented in eight different ways, corresponding to the relative position of the point [Sx,Sy] with respect to the point [Nx,Ny]. The symbol inequality of each linear term depends on the orientation of the RS. Linear terms (2) and (3) correspond to the case in which Nx>Sx and Ny>Sy. A summary of the study of the symbol inequality corresponding to each one of the eight different orientations of the RS is given in figure 5 (signs - and + correspond to symbol inequalities #< and #>, respectively).

6 Application

In this section a Qualitative Navigator Simulator will be described as an application of the model explained in the previous sections. The movement of an agent through the structured environment of the city of Castellón from an initial position to a goal, using as initial knowledge a set of qualitative spatial relationships, will be simulated.

The system architecture proposed to solve this problem consists of two modules: the Spatial Reasoning module and the Qualitative Navigator Simulator (QNavSim).

The Spatial Reasoning module contains a set of CSs, a CS for each one of the aspects of space which has been integrated in our model, and a set of extended objects processors, one for each spatial aspect considered. This module has been explained in the previous sections for cardinal directions. The QNavSim features will be introduced in this section.

The grid described in the previous section will be used here to represent the world model. The size of a unit of the grid which overlaps the city map of Castellón represents approximately the space which a person can walk in one minute. However, the reasoning process is also valid for any size of the grid unit. The world model consist of streets and landmarks, which are both approximated to intersections of the grid. Each portion of the most important streets of the city map of Castellón will be approximated by the closest straight line which goes from an intersection to a neighbor intersection of the grid. The information related to the streets is implemented in our approach by facts in the PROLOG database (i.e. *street (2,3,3,4)*). The name and position of the spatial landmarks are also defined as facts of the PROLOG database (i.e. *position(cinema,20,17)*). However, The position of landmarks are not known beforehand by the agent. When the agent arrives at each position, it checks the database for knowing if there is a landmark at this particular position. This represents the way in which the agent acquire new knowledge.

As in the rest of the paper, in the next example we are going to focus on cardinal directions. The initial information provided to the system will be the initial position in terms of the grid (La Paz square which is at (13,8)), the name of the goal (the bull ring) and the following set of cardinal direction relationships between landmarks of the city of Castellón: *(1) The Rey Don Jaime square is to the north-east or north or north-west of La Paz square; (2) Rey Don Jaime square is to the north-west, or north or north-east of Sol square; (3) Ribalta park is to the north-west or west of Rey Don Jaime square; (4) The Hospital is to the south or south-west of Ribalta park; (5) The bull ring is to the south of Ribalta park.*

Of course it is necessary that the agent knows the north-south direction. Using the previous set of cardinal direction relationships, it is possible to draw several cognitive maps which satisfy all the constraints (one of them is shown in figure 6). The fact that there is no unique solution which satisfies all the constraints means that the problem is not solved using only the reasoning process. It will be also necessary to simulate the movement of the agent through the streets. In its movement the agent will acquire new knowledge when it arrives to a new landmark. This new knowledge will be considered a stronger constraint to be added to the system that might reduce the goal region.

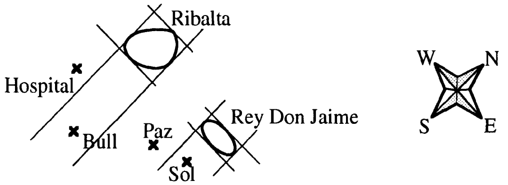


Figure 6. A cognitive map which satisfies all the cardinal relationships of the example.

	(c1)	ctr(1,Ribalta,Bull,[s],1)
	(c2)	ctr(6,Hospital,Bull,[n,ne,e,se,s,o],2)
ctr(Paz,Rey_Don_Jaime,[ne,n,nw])	(c3)	ctr(3,Rey_Don_Jaime,Bull,[nw,w,sw],2)
ctr(Sol,Rey_Don_Jaime,[nw,n,ne])	(c4)	ctr(2,Rey_Don_Jaime,Ribalta,[nw,w],1)
ctr(Rey_Don_Jaime,Ribalta,[nw,w])	(c5)	ctr(3,Paz,Ribalta,[ne,n,nw],2)
ctr(Ribalta,Hospital,[s,sw])	(c6)	ctr(3,Sol,Ribalta,[ne,n,nw],2)
ctr(Ribalta,Bull,[s])	(c7)	ctr(3,Paz,Rey_Don_Jaime,[ne,n,nw],1)
	(c8)	ctr(3,Sol,Rey_Don_Jaime,[ne,n,nw],1)
	(c9)	ctr(2,Ribalta,Hospital,[s,sw],1)
	(c10)	ctr(3,Rey_Don_Jaime,Hospital,[nw,w,sw],2)

a) b)

Figure 7. a) The initial constraints derived from the example; b) The constraints inferred by the Spatial Reasoning module.

The previous set of cardinal direction relationships are first of all translated into the set of constraints given in figure 7a). This set of constraints is the entry to the spatial reasoning module which will give back the set of inferred constraints shown in figure 7b).

The reasoning process is only executed once, at the beginning. The rest of the algorithm will interpret this information to guide the agent through the streets. If there is no street intersection at the current position, the agent will move straight on. If there is a street intersection, a choice has to be taken. If the agent is out of the goal or subgoal region, it will move towards that region. Otherwise, a subobjective is tried to be chosen to go towards it. In the case in which the agent is already inside the goal or subgoal region, it goes towards the street which is further away of the boundaries of the region in an attempt to stay inside the region more time. If no subgoal has been found, the agent gets lost. Each time the agent moves one step, it is checked if there is a landmark at the current position. If the landmark is the goal, the algorithm finishes, giving back the path followed by the agent. If the landmark is not the goal, it is checked whether there is a constraint which relates the current position with the goal (or in its defect a subgoal). If it exists, then the goal region might be further reduced. And again, if there is no constraint which relates a landmark —which has not visited yet— with another landmark —from the ones which have already been visited—, then the agent gets lost.

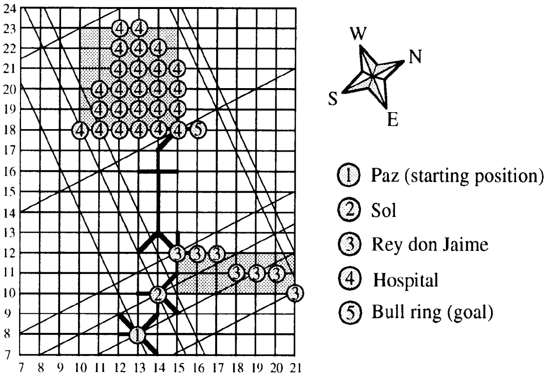


Figure 8. Path followed by the agent in the example.

In figure 8 the street intersections, the perceived landmarks and the trajectory of the agent, for this particular example, are shown. In the execution of the algorithm for this example, the following constraints have been used: (c6), (c3), (c4) and (c2).

6 Conclusions and future work

The main contributions of this work are the following: (1) The treatment of cardinal directions as a Constraint Satisfaction Problem will allow the integration of cardinal directions, orientation and distance into the same model. That means that any of these types of spatial relationships between the same spatial landmarks can be used in an uniform way to locate them. The programming paradigm CLP extended with CHRs provides a suitable level of abstraction for implementing a path consistency algorithm to propagate cardinal directions; (2) CLP(FD) provides a more restricted discrete environment where cardinal directions are solved more efficiently. CHR(FD) also provides the tool for using both point and extended objects as primitive of reasoning; (3) The model has been applied to a Qualitative Navigation Simulator where the movement of an "agent" through the structured environment of a city using cardinal directions as initial information has been accomplished. Although the initial information is very imprecise and it does not provide a solution to the problem by itself, the acquisition of new knowledge by the agent in its movement through the environment provides stronger constraints which will further reduce the goal region. Insufficient initial information, however, might provoke that the agent gets lost at a street intersection.

It is also important to notice that the cardinal directions model is equivalent to the orientation model defined by Hernàndez [6] where the front region is oriented towards the main door of a room. Therefore, the CS associated to this model can be described in the same way as the CS for cardinal directions. See [19] for further details.

The following ideas are proposed as subject for future work: (1) Further integration of another aspects of space into the same framework. For instance the integration of qualitative treatment of distance and time which would manage the concept of velocity qualitatively; (2) The treatment of inconsistency in the initial constraints; (3) The integration in the same model of a hierarchical structure of different levels of granularity; (4) The study of a parallel computation of the algorithms to improve efficiency; (5) The implementation of our spatial model in a real mobile robot which is our current interest.

References

1. Guesgen, H.W., "Spatial reasoning based on Allen's temporal logic", Technical Report TR-89-049, International Computer Science Institute, Berkeley, 1989.
2. Jungert, E., "The Observer's Point of View: An Extension of Symbolic Projections", in International Conference GIS -From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning in Geographic Space. Volume 639 of Lectures Notes in Computer Science. Ed. Springer-Verlag, 1992.
3. Mukerjee, A. & Joe, G., "A Qualitative Model for Space". In 8th-AAAI, pag. 721-727, 1990.
4. Freksa, C., "Using Orientation Information for Qualitative Reasoning" *Proc. Int. Conf. GIS- From Space to Territory: Theories and Methods of Spatio-Temporal Reasoning*, Pisa, 1992.

5. Freksa, C., Zimmermann, K., "On the Utilization of Spatial Structures for Cognitively Plausible and Efficient Reasoning", *Proceedings of the IEEE Int. Conf. on Systems, Man and Cybernetics*, pp. 18-21, 1992.
6. Hernández, D., *Qualitative Representation of Spatial Knowledge.* Lecture Notes in AI, 804. Ed. Springer-Verlag, 1994.
7. Frank, A., "Qualitative Spatial Reasoning about Distance and Directions in Geographic Space". In Journal of Visual Languages and Computing 3, pag. 343-373, 1992.
8. Meiri, I., "Combining qualitative and quantitative constraints in temporal reasoning", in Proc. of the 7th National Conference on Artificial Intelligence, pag. 260-267, 1991.
9. Escrig, M.T., Toledo, F., "Qualitative Spatial Orientation with Constraint Handling Rules", in the Proceedings of European Conference of Artificial Intelligence, pag. 486-490, 1996.
10. Escrig, M.T., Toledo, F., "Enhancing Qualitative Relative Orientation with Qualitative Distance for Robot Path Planning", in Proceedings of the 8th IEEE International Conference on Tools with Artificial Intelligence, pag. 174-182, 1996.
11. Escrig, M.T., Toledo, F., "Managing Orientation and Distance on Extended Objects in Qualitative Spatial Reasoning", in the International Joint Conference on Artificial Intelligence, Workshop on Spatial and Temporal Reasoning 1997.
12. Escrig, M.T., Toledo, F., "The integration of Qualitative Orientation and Named Distances: Application to a Qualitative Navigation Simulator", in the International Journal on Artificial Intelligence Tools, vol 6, no 3 & 4, 1997.
13. Meseguer, P., "Constraint Satisfaction Problems: An Overview", in AICOM, Volume 2, No. 1, March 1989.
14. Kumar, V., "Algorithms for constraint satisfaction problems: A survey". AI Magazine, 13(1), pag. 32.44, 1992.
15. Frühwirth, T., "Constraint Handling Rules", in Lecture Notes in Computer Science 910, 1994.
16. Mackworth, A.K. and Freuder, E.C., "The complexity of some Polynomial networks consistency algorithms for constraint satisfaction problems", Journal of Artificial Intelligence 25: pag. 65-74, 1985.
17. P. Brisset et al., ECLiPSe 3.4 Extensions User Manual, European Computer-Industry Research Center, Munich, Germany, 1994.
18. Hentenryck, P. V., "Constraint Logic Programming". The knowledge Engineering Review, Vol. 6:3, pag. 151-194, 1991.
19. Escrig, M. T., "Integration of several aspects of space into the same model using CLP extended with CHRs in Qualitative Spatial Reasoning", PhD thesis, Universitat Jaume I, Castellón (Spain), May 1997.

Definition and Study of Lineal Equations on Order of Magnitude Models*

Núria Agell[1], Fernando Febles[2], Núria Piera[2]

LEA-SICA (European Associated Laboratory of Intelligent Systems and Advanced Control)

[1]Dept. de Mètodes Quantitatius,
ESADE-Univ. Ramon Llull
08034 Barcelona
agell@esade.es

[2]Dept. de Matemàtica Aplicada II,
FIB-Univ. Politècnica de Catalunya
08028 Barcelona

Abstract. In situations where there is a lack of quantitative data description, it is important to use equations where the operators and the coefficients are qualitative. In addition, the results obtained have to be the qualitative description of the quantitative information that we would obtain by using numerical models. In this paper we define and study qualitative linear equations by using qualitative operators consistent with $\mathbb{R}$ in a qualitative model of magnitude orders.

1 Introduction

The scientific community's increased interest in qualitative approximations at the beginning of the 80's was in part motivated by the difficulty in using numerical information data. In other words, in order to apply some models, it is necessary to have previously manipulated the information. This usually impose a false precision.

To solve this problem the QR has developed new models and tools able to work with qualitative descriptions without forcing the data to fit a predetermined numerical model [6], [9]. It has been shown that it is possible to obtain important results from strictly qualitative information. In this sense, it is necessary to obtain operators that are consistent with $\mathbb{R}$, and at the same time maintain the maximum accuracy.

In other words, by operating two quantities, a and b, described qualitatively, we aim to ensure that the result will be the most accurate possible qualitative description of the magnitude that we would have obtained with access to the numerical description of a and b. So when we ask for consistency with $\mathbb{R}$, what we want is a qualitative operator that is the qualitative expression of a real operator.

*Paper partially supported by the **CICYT** project: *(TETRACCIS).TAP95-0446.*

2 The Absolute Order of Magnitude Model

In this section we give a description of the absolute order of magnitude model, that is the framework of this paper. The model used is a generalisation of the model studied in [8] and [2]. The number of labels we choose for describing our reality depends on the characteristics of each problem. The n dimensional absolute order of magnitude model (OM(n)) is built via a real line partition in 2n+1 classes.

Each class is named *basic description* and is represented by a label in the set $S_1=\{N_n, N_{n-1}, N_{n-2}, ..., N_2, N_1, 0, P_1, P_2, ..., P_{n-2}, P_{n-1}, P_n\}$.

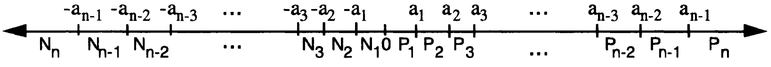

$-a_{n-1}$ $-a_{n-2}$ $-a_{n-3}$ ... $-a_3$ $-a_2$ $-a_1$ a_1 a_2 a_3 ... a_{n-3} a_{n-2} a_{n-1}
N_n N_{n-1} N_{n-2} ... N_3 N_2 N_1 0 P_1 P_2 P_3 ... P_{n-2} P_{n-1} P_n

Fig. 1.

Finally, the *quantity space* S is defined by extending S_1: for all $X, Y \in S_1$, with $X < Y$, the label [X,Y] is defined as follows :

$$[X,Y]=\begin{cases} X, & \text{if } Y=0; \\ Y, & \text{if } X=0; \\ \text{the smallest interval with respect to} \\ \text{the inclusion containing } X \text{ and } Y, & \text{if } X\neq 0 \text{ and } Y\neq 0. \end{cases}$$

On the other hand, it is possible to define an order relation $\leq_P$ in S, $\leq_P$ stands for less precise than", induced by the inclusion, graphically shown in figure 2.

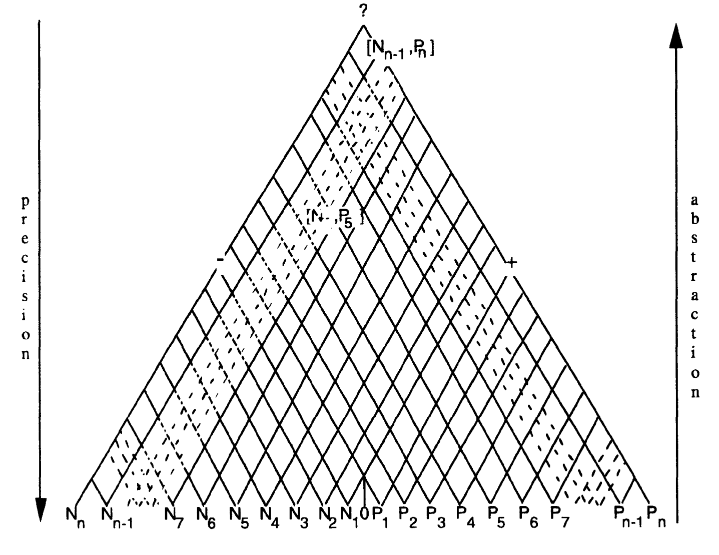

?
$[N_{n-1},P_n]$
$[N_4,P_5]$
-
+
precision
abstraction
N_n N_{n-1} N_7 N_6 N_5 N_4 N_3 N_2 N_1 0 P_1 P_2 P_3 P_4 P_5 P_6 P_7 P_{n-1} P_n

Fig. 2.

For all $X \in S$ the set $B_X = \{B \in S_1 - \{0\} : B \leq_p X\}$ is named *base of X*, and $B^*_X = \{B \in S_1 : B \leq_p X\}$ *enlarged base of X*. In an OM(n) q-equality is also defined. Given $X,Y \in S$ they are q-equal, or $X \approx Y$, if there is a $Z \in S$, such that $Z \leq_P X$ and $Z \leq_P Y$. This means they have a common basic element.

3 Qualitative expression of a real operator

Qualitativization of a real operator was introduced in [2]. The authors looked at the operators as functions defined between two different q-spaces. However, we consider the q-operators as bidimensional functions defined in an absolute order of magnitudes space.

When describing $\mathbb{R}^2$ in qualitative terms, what we consider is the Cartesian product S×S. Given a value in $\mathbb{R}^2$ its qualitative description is the qualitative description of both its coordinates.

We introduce the concept of qualitative expression of a real operator so that it can be used in the construction of qualitative linear equations. Given a real binary operator $\omega : \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}$, the *qualitative expression of ω*, is defined as the operator $\Omega : S \times S \longrightarrow S$ such that for all $X,Y \in S$, $\Omega(X,Y) = [\omega(X,Y)]$[1].

The following proposition has some important properties, which are satisfied by the qualitative expression of a real operator.

Proposition 1. Given $\Omega : S \times S \longrightarrow S$ the qualitative expression of a real operator $\omega : \mathbb{R} \times \mathbb{R} \longrightarrow \mathbb{R}$, then:

a) Ω conserves the precision and the q-equality in both components.

b) For all $X,Y \in S$ we have $\Omega(X,Y) = \left[\bigcup_{\substack{B_i \in B^*_X \\ B_j \in B^*_Y}} \Omega(B_i, B_j) \right]$, where B^*_X and B^*_Y are the enlarged bases of X and Y respectively.

The last condition is known as to be *generable from the base*. In other words we can obtain the result of $\Omega(X,Y)$ by operating the basic elements contained in X and Y.

Considering the definition given and an order of magnitude space defined via a symmetric partition of the real line given by the points $a,b \in \mathbb{R}^+$ in [4] the qualitative expression of the real sum was given, and later the same was done for the qualitative

[1] $\omega(X,Y) = \{z \in \mathbb{R};\ z = \omega(x,y)\ x \in X \text{ and } y \in Y\}$, and $[\omega(X,Y)]$ is the smallest element of S with respect to the inclusion containing $\omega(X,Y)$.

expression of the real product. In the case of q-sum there are three different tables depending on b<2a, b=2a or b>2a. In the case of the q-product we get nine different tables depending on b<1 where we distinguish $b^2<a$, $b^2=a$, or $b^2>a$, a>1 where we consider $a^2<b$, $a^2=b$, $a^2>b$ or finally b=1, a=1, and a<1<b.

Once we have the q-sum and the q-product defined on an order of magnitude space, we need to define an external product. If we take into account the fact that the qualitative description of a real number is a label on S, we will see the relation between both products.

Definición 1. Given a q-space (S,≈),of magnitude orders, associated to a symmetric partition of the real line, we define an external product $\odot: \mathbb{R}\times S \longrightarrow S$ such that for all $\lambda\in\mathbb{R}$ and $X\in S$, $\lambda\odot X=[\{z\in\mathbb{R}$ *where* $z=\lambda\cdot x$, *with* $x\in X\}]$.

The same symbol has been used for the external and internal products. We will see in each case if we are operating two elements from S or one from ℝ and one from S. In proposition 2 we give some properties related to the external and the internal product.

Proposition 2.

a) $\lambda\odot X\leq_P[\lambda]\odot X$ for all $X\in S$ and $\lambda\in\mathbb{R}$;

b) If $X\leq_P Y$, then $\lambda\odot X\leq_P\lambda\odot Y$ for all $X,Y\in S$ and $\lambda\in\mathbb{R}$.

4 Qualitative linear equations

Before formalising the *qualitative linear equations (QLE)*, let us first consider the resolution of the equation X⊕Y≈PL given on a OM(3).

Utilising the table for the q-sum[2], we obtain the solutions:

X	PM	PM	PL	PL	PL	PL	PL	PL	PL
Y	PS	PM	NL	NL	NS	0	PS	PM	PL

and vice-versa changing X by Y due to the commutativity of the q-sum operator.

In the example given, we have found, for instance, the solution X=PM and Y=PM, but if we consider X=[PS,PM] and Y=[PM,PL] they are also a solution. For this reason we will consider just the basic solutions, and use them to generate the rest of solutions.

Before introducing the qualitative equation concept, we must first solve the problem that arises from the lack of associativity of the qualitative operators. As is shown in example 1 when we have a q-equation with more than two unknowns, we get different solutions depending on the order in which the q-sum is done.

[2] It has been considered the q-sum operator when 2a<b.

Example 1. Let us consider the QLE given by:

$$X\oplus Y\oplus Z\approx NL^{2}$$

First, if we consider the equation in the form: $X\oplus(Y\oplus Z)\approx NL$, it has the q-solutions $X=PM, Y=NS$ and $Z=NM$ because $PM\oplus(NS\oplus NM)=PM\oplus[NL,NM]=[NL,PM]\approx NL$.

But if we consider the equation in the form $(X\oplus Y)\oplus Z\approx NL$, the q-values $X=PM, Y=NS$ and $Z=NM$ are not a solution because $(PM\oplus NS)\oplus NM=[PS,PM]\oplus NM=[NM,PM]$, and $[NM,PM]\not\approx NL$.

To solve this problem, we must consider the next definition when operating more than two S elements [1].

Definición 2 For any $(X_1,X_2,\ldots,X_n)\in S^n$, we define:

$$\overset{n}{\underset{i=1}{\oplus}}(X_1,X_2,\ldots,X_n)=[\{z\in \mathbb{R}/\ z=x_1+x_2\ldots+x_n \text{ with } x_i\in X_i\}].$$

The n-dimentional operator given is well defined and the value of $\overset{n}{\underset{i=1}{\oplus}}(X_1,X_2,\ldots,X_n)$ is univocaly determinate because the real sum is an associative operator. We are now in a position to give the definition of qualitative linear equation.

Definición 3 Let $(S,\approx)$ be an order of magnitude qualitative space, a linear qualitative equation with variables X_1, $X_2,\ldots$, X_n, is an equation such as: $A_1\odot X_1\oplus A_2\odot X_2\oplus\ldots\oplus A_n\odot X_n\approx C$, where $C\in S$ and for all $i\in\{1,\ldots,n\}$, $A_i\in S$ or $A_i=\lambda_i\in\mathbb{R}$.

The confluences defined by [3] and studied by many authors [9] are a particular case of QLE, when the coefficients are given in the sign algebra. The next proposition ensures the existence of a solution for any qualitative linear equation.

Proposition 3. Given a linear qualitative equation it has basic solutions.

This is demonstrated by constructing a real linear equation from the QLE given, and when the real equation is solved, we give its qualitative description and it is a solution of the QLE given.

Example 2. Given the equation: $+\odot X_1\oplus NL\odot X_2\approx PM$, defined in an OM(3) where the real values that give the real line partition are $a_1=2$ and $a_2=5$. The real equation can then be established as $x_1-6x_2=3$, from which a pair of solutions are: $x_1=3$ and $x_2=0$.

We then consider the qualitative expression of the real solutions and obtain $X_1=PM$ and $X_2=0$. It is easy to prove that these q-values are a solution, because $+\odot PM\oplus NL\odot 0=+\approx PM$.

It is shown that every qualitative linear equation has solutions in S, but what we want to know are all the qualitative vectors $(\overline{X}_1, \overline{X}_2, \cdots, \overline{X}_n) \in S^n$, that are solution of the equation $A_1 \odot X_1 \oplus A_2 \odot X_2 \oplus \ldots \oplus A_n \odot X_n \approx C$.

We must realise that if a pair of q-values (X,Y) are a solution of a QLE then any pair of q-values less precise than (X,Y) will be a solution. In proposition 4 we give a method to generate new solutions from a solution given.

Proposition 4. Given the QLE $A_1 \odot X_1 \oplus A_2 \odot X_2 \oplus \ldots \oplus A_n \odot X_n \approx C$, if $(\overline{X}_1, \overline{X}_2, \cdots, \overline{X}_n) \in S^n$ is a solution, then for all $(Y_1, Y_2, \cdots, Y_n) \in S^n$ such that $\overline{X}_1 \leq_P Y_1, \overline{X}_2 \leq_P Y_2, \cdots, \overline{X}_n \leq_P Y_n$, we have that $(Y_1, Y_2, \cdots, Y_n)$ are solution of the given equation.

Proof:

As the q-sum and the q-product are the qualitative expressions of real operators, proposition 1 ensures that they conserve the precision, so

$$A_1 \odot \overline{X}_1 \oplus \cdots \oplus A_n \odot \overline{X}_n \leq_P A_1 \odot Y_1 \oplus \cdots \oplus A_n \odot Y_n,$$

On the other hand $(\overline{X}_1, \overline{X}_2, \cdots, \overline{X}_n)$ is a solution of the given equation, we then have:

$$A_1 \odot \overline{X}_1 \oplus \cdots \oplus A_n \odot \overline{X}_n \approx C.$$

We get an element of S: $A_1 \odot \overline{X}_1 \oplus \cdots \oplus A_n \odot \overline{X}_n$ that is more precise than $A_1 \odot Y_1 \oplus \cdots \oplus A_n \odot Y_n$, and qualitatively equal to C, and for this reason we have $A_1 \odot Y_1 \oplus \cdots \oplus A_n \odot Y_n \approx C$ ■

We will now show that solving qualitative equations requires us to search for the basic solutions, which all other solutions will be obtained.

Proposition 5. Given the QLE $A_1 \odot X_1 \oplus A_2 \odot X2 \oplus \ldots \oplus A_n \odot X_n \approx C$, with $(Y_1, Y_2, \cdots, Y_n) \in S^n$ solution, there exist $(\overline{X}_1, \overline{X}_2, \cdots, \overline{X}_n) \in S_1^n$ solution of the linear equation given.

Proof:

As $(Y_1, Y_2, \cdots, Y_n) \in S^n$ is a solution, we have:

$$\bigoplus_{i=1}^{n} (A_1 \odot Y_1, A_2 \odot Y_2, \ldots, A_n \odot Y_n) \approx C,$$

i.e. $[M] \approx C$,where,

$$M = \{z \in \mathbb{R} / \ z = a_1 y_1 + a_2 y_2 \ldots + a_n y_n \text{ with } a_i \in A_i \text{ i } y_i \in Y_i\}.$$

For this reason there exists a basic element $B \in [M] \cap C$.

We can consider the vectors $(y_1,y_2,\cdots,y_n)\in Y_1\times Y_2\times\ldots\times Y_n$ and $(a_1,a_2,\cdots,a_n)\in A_1\times A_2\times\ldots\times A_n$ that satisfies the next condition: $[a_1\cdot y_1+a_2\cdot y_2+\ldots+a_n\cdot y_n]=B\leq_P C$. Then taking $(\overline{X}_1,\overline{X}_2,\cdots,\overline{X}_n)\in S_1^n$ such that $[y_i]=\overline{X}_i$ for all $i\in\{1,2,\ldots,n\}$ we get a basic solution from the QLE given.

Finally, from proposition 5 we will give a theorem that will permit us to easily solve the QLE by translating them to equations where the coefficients are basic elements.

Theorem 1. Given the QLE $A_1\odot X_1\oplus A_2\odot X_2\oplus\ldots\oplus A_n\odot X_n\approx C$, then $(\overline{X}_1,\overline{X}_2,\cdots,\overline{X}_n)\in S_1^n$ is a solution if, and only if, there exist $B_1,B_2,\ldots,B_n$ and B basic elements with $B_i\in B^*_{A_i}$ for all $i\in\{1,\ldots,n\}$ and $B\in B^*_C$, such that $(\overline{X}_1,\overline{X}_2,\cdots,\overline{X}_n)$ is a solution of the QLE $B_1\odot X_1\oplus B_2\odot X_2\oplus\ldots\oplus B_n\odot X_n\approx B$.

5 An application to qualitative calculus

At this point we will solve a q-equation system given by the classic problem of the impact of two masses. We study the example given in [7], and we want to predict the outcome of an elastic collision between a small and a big mass m and M coming from opposite directions at speeds v_i and V_i.

We consider an order of magnitudes model defined from a real line partition on seven classes. Where the points $a, b\in\mathbb{R}^+$ defining the partition satisfies $1<a<a^2<b$ and $2a<b$.

First we consider the real equations, the quantitative model. The collision is represented by the conservation equations of quantity of movement and kynetic energy:

$$\left.\begin{array}{l} mv_i+MV_i=mv_f+MV_f \\ \frac{1}{2}mv_i^2+\frac{1}{2}MV_i^2=\frac{1}{2}mv_f^2+\frac{1}{2}MV_f^2 \end{array}\right\}$$

This can be reduced to:

$$\left.\begin{array}{l} mv_i+MV_i=mv_f+MV_f \\ v_i+v_f=V_i+V_f \end{array}\right\}$$

Changing the real coefficients, variables and operators for their qualitative representation, and equality for qualitative equality, we obtain the following system of linear qualitative equations:

$$\left.\begin{array}{l} m\otimes v_i\oplus M\otimes V_i\approx m\otimes v_f\oplus M\otimes V_f \\ v_i\oplus v_f\approx V_i\oplus V_f \end{array}\right\}$$

We analyse the case where the mass m is negligible in respect to M and m has a negligible speed in respect to M, in the sense that was defined in [5]. This is shown in the following drawing.

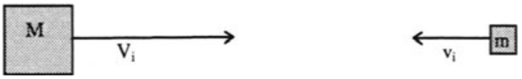


Fig. 3.

From the fact that v_iNeV_i, and they are in opposite directions we find that v_i=NS and V_i=PL or v_i=PS and V_i=NL. We will solve only the case v_i=NS and V_i=PL due the symmetry of both cases.

On the other hand, as the masses must take positive values we consider M=PL and m=PS.

To sum up, we must search for two q-values v_f and V_f such that they satisfy:

$$\left.\begin{array}{l} PP \otimes NP \oplus PG \otimes PG \approx PP \otimes v_f \oplus PG \otimes V_f \\ NP \oplus v_f \approx PG \oplus V_f \end{array}\right\}$$

When we consider only the second QLE we get the 16 basic values for v_f and V_f given in the following table:

vf	NL	NM	NS	0	PS	PS	PM	PM	PM	PL	PL	PL	PL	PL	PL	PL
Vf	NL	NM	NL	NL	NL	NM	NL	NM	NS	NL	NM	NS	0	PS	PM	PL

Table 1.

This result is independent from the q-value taken by m and M. On the other hand if we now consider the first QLE given in the system it can be reduced to:

$$+ \approx PP \otimes v_f^{`} \oplus PG \otimes V_f .$$

And when we substitute the q-values found in table 1, they can be reduced to the 10 basic solutions shown in table 2:

v_f	PM	PM	PM	PL	PL	PL	PL	PL	PL	PL
V_f	NL	NM	NS	NL	NM	NS	0	PS	PM	PL

Table 2.

Similar attempts to solve the problem, were made with a finer partition of the real line. We can see that when the dimension of the model increases, we obtain the same results. For example, if we consider an order of magnitude model, given by 9 basic elements, i.e. $S_1=\{N_4,N_3,N_2,N_1,0,P_1,P_2,P_3,P_4\}$[3], we obtain the following solutions:

v_f	P_3	P_4	P_3	P_4	P_3	P_4	P_3	P_4	P_4	P_4	P_4	P_4	P_4
V_f	N_4	N_4	N_3	N_3	N_2	N_2	N_1	N_1	0	P_1	P_2	P_3	P_4

Table 3.

It is important to realise that the sign of v_f is always positive, and that globally the pairs of solutions are exactly the same as suggested by Commonsense.

6 Conclusions

The results presented permit one to pass a linear equation to a qualitative equation, in order to solve problems where quantitative data is not available.

On the other hand, it is possible to join the results obtained to some qualitative binary relations, such as "being negligible of" or "having the same order of magnitude", in order to improve the techniques for solving qualitative linear systems.

From the studied techniques we are constructing a sofware tool able to build an order of magnitude model and there it will permit to define and solve qualitative lineal equations.

References

1. Agell, N., Piera, N.: The Associative Problem for Qualitative Operators. In: Proceedings of the 14th IMACS World Conference, (1995) pp.1413-1415.

2. Missier, A., Piera, N., Travé, L.: Order of Magnitude Algebras: a Survey. In: Revue d'Intelligence Artificielle, no. 4, vol. 3, (1989) pp. 95-109.

3. Kleer de J., Brown, J.S.: A Qualitatative Physics Based on Confluences. In: Artificial Intelligence 24, (1984) pp. 7-83.

4. Piera, N., Sánchez, M., Travé-Massuyès, L.: Qualitative Operators for Order of Magnitude Reasoning: Robustness and Precision. In: Proceedings 43th IMACS World Congress on Computation and Applied Mathematics, Dublin (1991), pp.1-6.

5. Piera, N., Agell, N.: Binary Relations for Qualitative Calculus. In Proccedinings of the IEEE-SMC International Conference, Chicago (1992) pp. 267-271.

6. Piera, N (ed.): Current Trends in Qualitative Reasoning and Applications. Ed. CIMNE, Monograph nº 33, Barcelona (1995).

7. Raiman, O.: Order of Magnitude Reasoning. Artificial Intelligence, núm. 51 (1991) pp. 11-38.

[3] the points $a_1,a_2,a_3\in\mathbb{R}^+$ defining the partition are: $a_1=2$, $a_2=5$, $a_3=12$.

8. Travé, L., Piera, N.: The Orders of Magnitude Models as Qualitative Algebras. 11th IJCAI, Detroit (1989) pp. 1261-1266.

9. Travé-Massuyès, L., Dague,Ph., Guerrin, F.: Le raisonement qualitatif pour les sciences de l'ingénieur. Ed. Hermes, Paris(1997).

A Constraint-Based Approach to Assigning System Components to Tasks*

Elise H. Turner and Roy M. Turner

Department of Computer Science, University of Maine, Orono, ME 04469, USA
{eht, rmt}@umcs.maine.edu

Abstract. In multi-component systems, individual components must be assigned to the tasks that they are to perform. In many applications, there are several possible task decompositions that could be used to achieve the task, and there are limited resources available throughout the system. We present a technique for making task assignments under these conditions. Constraint satisfaction is used to assign components to particular tasks. The task decomposition is selected using heuristics to suggest a decomposition for which an assignment can be found efficiently. We have applied our technique to the problem of task assignment in systems of underwater robots and instrument platforms working together to collect data in the ocean.

1 Introduction

In this paper, we address the problem of assigning tasks to components of systems comprised of multiple robots and instruments. Our method views each component as a collection of capabilities. Each component is also expected to have limited resources which can be expended during the mission. The overall mission of the system is decomposed into subtasks which are further decomposed into the capabilities which are required to achieve them. Task assignment is viewed as a constraint satisfaction problem in which the assignment of a component to a particular task is constrained by the capabilities and resource limitations of the component.

Our method has been developed specifically for *Autonomous Oceanographic Sampling Networks (AOSNs)* [1]. AOSNs are multi-component systems that will be deployed for long durations to collect data in the ocean. We will call specific high-level tasks *missions* of the system to distinguish them from the subtasks which will be assigned to individual components. Components of the AOSN include both underwater Vehicles and non-mobile Instrument Platforms (VIPs). Because of the limited number of underwater vehicles in existence, and because of the expense of developing new VIPs, the AOSN should be able to take advantage of slack resources in the ocean engineering community. Consequently, the configuration of the system will be driven by the availability of components.

* This work was funded in part by contract N0001–14–96–1–5009 from the Office of Naval Research. Thanks to Dick Blidberg, Steve Chappell, Jim Kadin, and the UMaine CDPS group for many insightful comments on this work.

AOSNs share several important characteristics with other multi-component systems. These include:

1. Each component will have a collection of capabilities, usually many more than one. However, not all components have the capability to reason or to act autonomously in the world.
2. Because of resource limitations, not all components will be able to contribute all of their capabilities to the mission.
3. There may be some special-purpose components which have rare capabilities that are needed for the mission.
4. The total number of resources and capabilities in the system may be close to the total needed to perform the mission.
5. The task will have several known decompositions, each of which will require a different combination of capabilities.
6. The system must be able to organize itself autonomously. This may be necessary for the initial organization or for the system to respond to changes in the user's goals or in the environment.

In the next section, we describe our technique in more detail. We present an example in Sect. 3. In Sect. 4, we compare our work to related systems. We conclude with a discussion of our results and plans for future work.

2 Assigning Components to Tasks

We assume that each mission can be achieved using one of several *task decompositions.* A task decomposition breaks down a task into subtasks, then those into their subtasks, etc. Each task decomposition requires different capabilities and demands different levels of resources. The problem of task assignment can be seen as having two parts: selecting the appropriate task decomposition and assigning VIPs to the capabilities required for that alternative. Since the discussion of selecting the decomposition requires an understanding of how assignment is done, we present that first.

2.1 Assigning VIPs to Capabilities

The assignment of VIPs to capabilities can be represented as a constraint satisfaction problem in a straightforward way. The variables in the constraint satisfaction problem are the capabilities required. The value of each variable is the VIP which is assigned to contribute that capability to the task. There are three constraints on this assignment:

1. The VIP assigned to the variable must have the required capability.
2. The VIP must have the required resources available.
3. The total resources required by all of the capabilities to which the VIP is assigned cannot exceed the VIP's available resources.

The first two are unary constraints. They are checked when VIPs are assigned to the initial domains of the variables, which are created when the task decomposition tree is formed. The third is an n-ary constraint between all variables which have the given VIP in their domains. There will be one such constraint for each VIP.

We use the *constrained heuristic search* (CHS) [2] formalism for solving the constraint satisfaction problem. CHS combines constraint satisfaction and heuristic search. Search states are *constraint graphs.* As in standard CSPs, variables are represented as nodes and their domains are their remaining potential values. In CHS, constraints are also represented by nodes. Constraint nodes are adjacent to the variable nodes whose values they restrict. This allows constraints to be added to the constraint graph after some constraint propagation has taken place. The topology of the constraint graph can be characterized by a set of *textures.* Heuristics are developed to approximate these textures. The heuristics can then be applied to select the operators used to generate the next search state. Constraints are propagated within each newly created state. Any search mechanism can be applied to the search space.

The heuristics that we apply during constraint propagation select the unassigned variable with the fewest values left in its domain and assign to it the value that is a member of the fewest domains. The state space is searched using a version of hill-climbing which allows backtracking to previous choice points.

2.2 Selecting the Task Decomposition

We assume that the VIPs, their capabilities and resource limitations, and the task decomposition tree for the mission are known before task assignment begins [3]. The task decomposition tree has the form shown in Fig. 1. The subtasks that must be performed to accomplish a task are the ANDed children of that task. In the figure, T1 and T2 must be performed in order to perform the mission. Alternative methods for performing a task are the ORed children of the parent task. For example, Alt-T1-1 and Alt-T1-2 are two alternatives for performing T1. All of the children of the node are either ANDed or ORed together; there are no mixed nodes. The children of the root can be either ANDed or ORed together. The leaves of the task decomposition tree are variable nodes representing capabilities required to achieve the parent task. These nodes can be moved to the constraint graph. The capabilities are shown in our figure by letters. We have numbered the capabilities which appear more than once to indicate that they are distinct usages of the same capability and to identify the node in our discussion of the example. In addition to the capability, the number of "resource units" required[1] and the domain of the variable are associated with these nodes. The task decomposition tree represents the ways in which the mission can be performed. For our example, the mission can be performed using the following collections of capabilities: {A, B1, D2}; {A, B1, B2, E}; {C, D1, D2}; and {C, D1, B2, E}.

[1] These correspond to time, energy, etc.

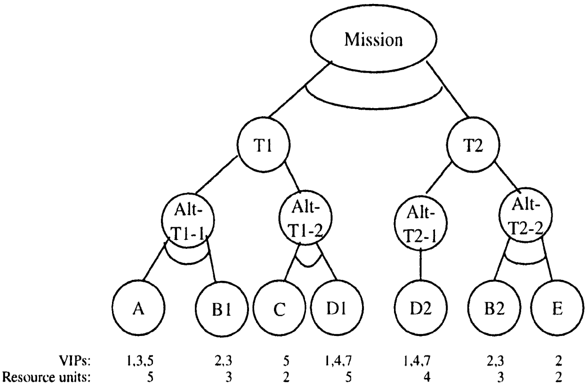


Fig. 1. A task decomposition tree for a mission

To build the constraint graph as described above, we must add all of the capabilities which constitute a decomposition to the constraint graph. Capabilities are selected using textures from CHS so that the resulting constraint graph can be solved efficiently. In particular, we focus on textures that increase the likelihood that a solution can be found without backtracking.

Our algorithm starts with an empty constraint graph and the initial task decomposition tree; this is the initial state. A new state is generated by selecting a leaf node or ANDed leaf nodes to add to the graph. Each selection represents a partial commitment to one particular task decomposition. Selections are made by applying two recursive algorithms (Fig. 2), *select-alternative* at OR nodes and *select-subtask* at AND nodes, starting at the root of the tree and ending when a selection is made at the leaf level.

```
select-alternative(node):
  if node is leaf then
    return(size(domain),node);
  else
    call select-alternative on each child;
    if node is OR then
      return(max value, child with max value);
    else ;; it's an AND node
      return(min value, child with min value);
    fi;
  fi;
end;
```

```
select-subtask(node):
  if node is leaf then
    return(size(domain),node);
  else
    call select-subtask on each child;
    if node is AND then
      return(min value, child with min value);
    else ;; it's an OR node
      return(sum of values of children);
    fi;
  fi;
end;
```

Fig. 2. Algorithms used to select capabilities

Following the *value goodness* texture [2], in *select-alternative* we choose the alternative that can be satisfied by the greatest number of values. This gives the constraint algorithm the most flexibility. Since the VIPs that can perform the task are the potential values for the variables, they provide the basis for the selection. The values are propagated up the tree from the leaves so that the interdependence of the ANDed siblings can be taken into account.

Following the *constraint reliance* texture [2] in *select-subtask* we choose the subtask with the fewest possible alternatives. Since this subtask is the most constrained, it needs to be added to the constraint graph early. This ensures that these important constraints will be in the constraint graph, and will be used to construct partial solutions, early in problem solving.

When a node is selected, its associated capabilities are added to the constraint graph. The task decomposition tree is then pruned to remove the selected nodes and other nodes which cannot be part of the same task decomposition. This includes pruning alternatives to ancestors of the node.

3 Example

Returning to the task decomposition in Fig. 1, we will now consider how the technique described above would select a task assignment. This tree is reproduced in Fig. 3 showing the domain of each variable and resource units required. Suppose that there is the following situation with respect to VIPs: VIP1 has capabilities A and D, with 6 resource units total; VIP2 has B and E, with 3 units; VIP3 has A and B, with 5 units; VIP4 has D, with 5 units; VIP5 has A and C, with 5 units; VIP6 has no capabilities needed for this task; and VIP7 has D, with 5 units.

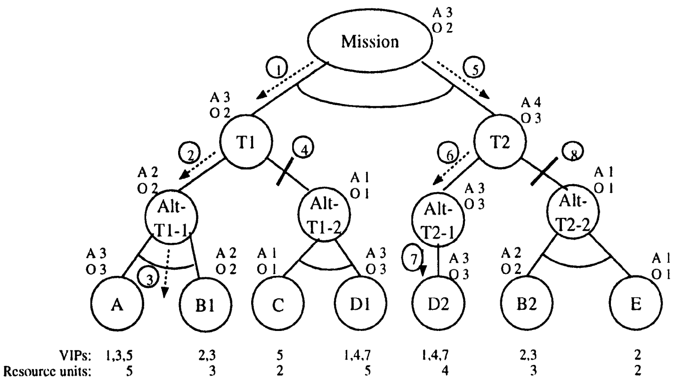


Fig. 3. Annotated task decomposition tree

The numbers beside the nodes labeled with "A" and "O" show the and- and or-ratings of the node, respectively, as computed by *select-alternative* and *select-*

subtask.[2] The progress of the capability selection algorithm is shown on the tree. Arrows indicate selections of children and slashes indicate branches that have been pruned. These marks are numbered and the corresponding number appears in parentheses after the text describing that step.

Starting with the root of the marked tree, *select-subtask* chooses the subtask to consider first. It selects the subtask with the lowest and-rating, T1 (1). From there, *select-alternative* will select the child of T1 with the highest or-rating, Alt-T1-1 (2). This alternative has only ANDed children, so they become the first variables added to the constraint graph (3). The first part of Fig. 4 shows the constraint graph after A and B1 are added. Double circles show the n-ary resource constraints, labeled with the VIP concerned and resource limit. Constraints are checked at this point, but since no commitment has yet been made to a value for either variable, no values are eliminated from either domain. The tree is now pruned (4) to eliminate non-chosen OR branches in the ancestors. At this point, a new search state has been created.

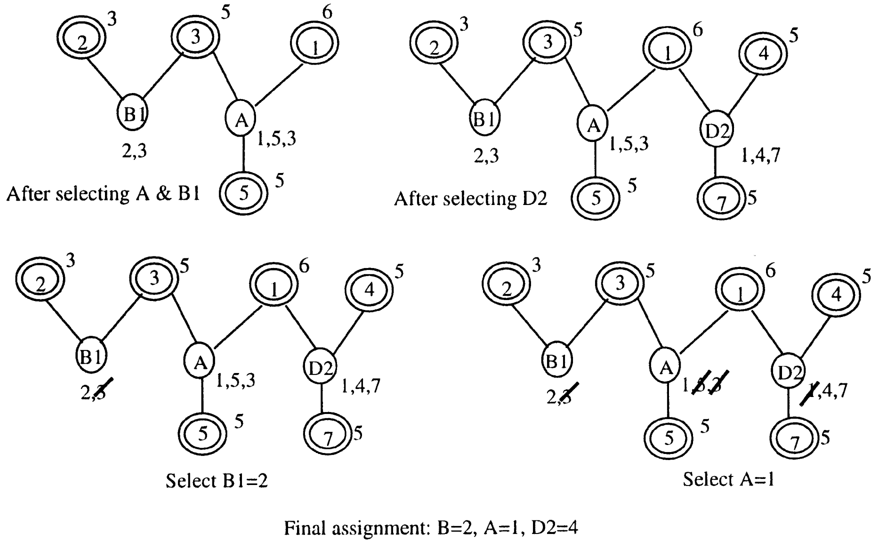


Fig. 4. Evolution of the constraint graph in succeeding states

From this state, the algorithm returns to the top of the tree and selects the only remaining subtask, T2 (5). *Select-alternative* now selects Alt-T2-1, and D2 is added to the constraint graph, as shown in the upper right of Fig. 4. The constraint graph is consistent, the tree is pruned (7), and another search state is created.

[2] An optimization is done in the program to record the markings at the nodes so that *select-subtask/-alternative* do no unnecessary work.

At this point, there are no more subtasks at the top level, so a complete possible configuration is represented in the constraint graph. The normal CHS algorithm now takes over. As shown in the bottom of Fig. 4, CHS selects the variable (B2) with the fewest values in its domain, then assigns to it the value (VIP2) that appears in the fewest domains. Constraint propagation does not prune any other domains. From this state, CHS assigns VIP1 to A. Constraint propagation prunes D2's domain in this case. Our heuristics evaluate all remaining values for D2 as equal, so VIP4 is arbitrarily chosen for the value of D2. This yields a goal state, A=VIP1, B1=VIP2, D2=VIP4.

Although space does not permit us to include backtracking in our example, backtracking could occur under two circumstances. The system can backtrack during constraint satisfaction to try different choices for assigning values to VIPs. If the constraint graph cannot be satisfied, the system can backtrack to states created during capability selection to try a different task decomposition.

4 Related Work

Several techniques from distributed artificial intelligence can be used to do task assignment (e.g., [4], [5]). However, most of them rely on local decisions, often made by local negotiation between agents, to produce global solutions. This can lead to sub-optimal solutions or even to no solution at all, since there is no global perspective and no possibility to coerce an agent into doing a locally-unacceptable, but globally-necessary, task. It is difficult to see how a technique such as the Contract Net Protocol [4] or Partial Global Planning [5] can handle selecting between alternative configurations; this requires a global notion of what the possible configurations are. The local contracts or agreements in those systems would also over-commit to alternatives that are tentative. Some approaches also require sophisticated agents [5], which we cannot guarantee in our application.

Our technique is also related to applications of CHS in other domains. At first glance, job-shop scheduling [6] appears to be the most closely-related application because it is concerned with scheduling resources to perform tasks. However, it differs from our domain because there are no alternative decompositions for performing the task.

Other applications of CHS have considered alternative constraint graphs. Of these, Wright [7] and CORAL [8] are most closely related to our work. Wright uses texture-based heuristics to select alternative layouts for spatial planning problems. However, Wright relies on representing possible alternatives in the CHS formalism for constraint graphs instead of in a standard task decomposition tree. The CHS formalism is difficult to use in our domain and is less than ideal for any application in which the task decomposition tree will be provided by some other planner.

CORAL works in the domain of configuring and allocating available components to assist inventory managers. It represents possible configurations in a standard task decomposition tree, selects a feasible configuration, and assigns resources from its inventory to the configuration. CORAL is very similar to our work. First, selections are made from the task decomposition tree, propagating constraints when relevant. After a complete configuration is selected, assignments are made to actual components.

CORAL differs from our work in two significant ways. First, CORAL does not take advantage of texture-based heuristics when selecting its configuration. We have found that using such heuristics increases the overall efficiency for task assignment [9]. Second, CORAL selects a particular type of part that will be used in a configuration from the task decomposition tree. In our domain, it is analogous to selecting the VIP that will be used to perform a task as part of choosing the alternative. In other words, this would be analogous to binding a variable to a value before adding the variable to the constraint graph. The technique works in CORAL's domain because there are usually enough parts available to make any configuration. However, in our application, texture heuristics must be used to help select VIPs for task assignment because not all assignments will be possible. Consequently, the resource limitations of the VIPs must be taken into account when a VIP is assigned to a task.

We have implemented a version of CORAL, adapting it to our application by assigning a VIP when a capability is selected from the task decomposition tree. This version of CORAL was used to collect the data described below. Our early experiements with CORAL suggest that our technique reduces backtracking for our application.

5 Conclusion

Our system was used to produce task assignments for 100 randomly-generated, solvable task decomposition trees. Examples in this set had a mean of 2574 configurations (possible task decompositions) per mission. The median time of solution was 50 ms.[3] The median number of backtracks per runs in which the system backtracked was 1. Our version of CORAL was run on the same set of task decomposition trees as above. It had a median run time of 70 ms and backtracked a median of 1.5 times per run with backtracks.

Our technique will be used in our AOSN to make task assignments. In the future, we plan to run additional experiments to test our heuristics for selecting task decompositions. We also plan to integrate our operator for selecting capabilities with the other CHS operators.

References

1. Curtin, T.B., Bellingham, J.G., Catipovic, J., Webb, D.: Autonomous oceanographic sampling networks. Oceanography. **6** (1993)

[3] Since there was an outlier that skewed the mean, we are using median values. The constraint graph produced for the outlier had a high number of backtracks. A mechanism for identifying constraint graphs which will have high numbers of backtracks and selecting values early on to avoid backtracking is discussed in [9].

2. Fox, M.S., Sadeh, N., Bayken, C.: Constrained heuristic search. In: Eleventh International Joint Conference on Artificial Intelligence. (1989) 309 - 315
3. Turner, R.M., Turner, E.H.: Organization and reorganization of autonomous oceanographic sampling networks. In: International Conference on Robotics and Autonmation. (to appear)
4. Smith, R.G.: The contract net protocol: High-level communication and control in a distributed problem solver. IEEE Transactions on Computers. **12** (1980) 1104 - 1113
5. Durfee, E.H., Lesser V.R.: Negotiating task decomposition and allocation using partial global planning. In: Gasser, L., Huhns, M. N. (eds.): Distributed Artificial Intelligence. Morgan Kaufmann Publishers, Inc., San Mateo, CA (1989) 229 - 243
6. Sadeh, N., Fox, M.S.: Variable and value ordering heuristics for the job shop scheduling constraint satisfaction problem. Artificial Intelligence. **86** (1996) 1 - 41
7. Baykan, C.A., Fox, M.S.: Constraint techniques for spatial planning. In: ten Hagen, P.J. W., Veerkamp, P. J. (eds.): Intelligent CAD Systems III: Practical Experience and Evaluation. Springer-Verlag, New York (1991) 187 - 204
8. Sathi, N., Fox, M.S., Goyal, R., Kott, A.S.: Resource configuration and allocation: A case study of constrained heuristic search. IEEE Expert. **7** (1992) 26 - 35
9. Turner, E.H., Turner, R.M.: Selecting task decompositions for constrained heuristic search (in preparation)

Automatic Semiqualitative Analysis: Application to a Biometallurgical System

R. M. Gasca, J. A. Ortega, M. Toro

Departamento de Lenguajes y Sistemas Informáticos,
Facultad de Informática y Estadística
Avda. Reina Mercedes s/n, Sevilla (España)
e-mail:{gasca,jaortega,mtoro}@lsi.us.es

Abstract. The aim of this work is the representation and analysis of semiqualitative models. Their qualitative knowledge is represented by means of qualitative operators and envelope functions. A semiqualitative model is transformed into a family of quantitative models.
In this paper the analysis of a model is proposed as a constraint satisfaction problem. Constraint satisfaction is an umbrella term for a variety of techniques of Artificial Intelligence and related disciplines. In this paper attention is focused on intervals consistency techniques. The semiqualitative analysis is automatically made by means of consistency techniques. The presented method is applied to a industrial biometallurgical system in order to show how increase the capacity of production.

1 Introduction

In engineering and science, the models made up for the study of dynamical systems are normally composed of quantitative and qualitative knowledge. This knowledge is composed by both of them. It is known as semiqualitative knowledge. Real models contain quantitative, qualitative and semiqualitative knowledge. All this knowledge must be considered when these models are studied.

The techniques developed to analyze and simulate quantitative models are well known. A great variety of techniques has been studied for the representation and the manipulation of qualitative knowledge, such as algebra of signs, interval arithmetic, fuzzy sets, and order of magnitude reasoning.

In order to analyze industrial models, it is necessary sometimes to solve conflicts on the request of accuracy and flexibility. The models of dynamical systems should provide different levels of numerical abstraction for their elements. These levels may be a purely qualitative description [8], semiqualitative [2], [6], numerical based on intervals [11], quantitative and mixed of all levels [7].

On other hand, the systems dynamics obtains the differential equations of a system from its structure. This technique could obtain different qualitative behaviors of a given structure. The analysis of these behaviors constitutes the qualitative analysis of dynamical systems. The mathematical qualitative theory of dynamical systems involved studying qualitatively the behaviour (e.g. asymptotic behaviour) of time evolving systems.

In order to automate the qualitative analysis of dynamic systems several applications have been developed. They combine techniques of numerical methods with symbolic computation, and methods proceeding from the knowledge of the science and the mathematics. These applications begin with the top-level specifications of physical model. They prepare simulation experiments, and accomplish them. Also they interpret the numerical results, and they formulate the results in qualitative terms. Among them, we can cite *PLR* [9], *bifurcation interpreter* [1], *KAM* [12], *POINCARE* [10], and *MAPS* [13]. In this paper, a method to carry out the analysis of dynamical systems automatically is shown. The semiqualitative analysis is proposed as a set of interval constraint satisfaction problems. They are solved applying consistency techniques [5].

2 Semiqualitative models

A dynamical system can be considered as the constraints

$$\Phi(\dot{x}, x, p), \qquad x(t_0) = x_0, \qquad \Phi_0(p, x_0) \tag{1}$$

being x the state variables of the system, p the parameters, $\dot{x}$ the variation of the state variables with the time, Φ_0 the constraints among parameters and initial conditions, and Φ the constraints on $\dot{x}, x$ and p. The dynamical system represented in (1) can symbolically be transformed into a set of contraints with variables, parameters and intervals. In this paper, we only study systems that can be transformed as

$$\dot{x} = \tilde{f}(x, p), \qquad x(t_0) = x_0, \qquad \Phi_0(p, x_0) \tag{2}$$

The vector field $\tilde{f}$ may be composed of quantitative and qualitative variables, constants, arithmetic operators, functions and envelope functions, expressed as it is indicated in our previous paper [4], where qualitative variables and envelope functions are transformed to interval expressions. If we take into account the stablished concepts in that paper, the dynamical system (2) is transformed in

$$\dot{x} = f(x, r, p), \qquad x(t_0) = x_0, \qquad \Phi_0(p, r, x_0) \tag{3}$$

where $r \in \mathbb{I}$ are new parameters, $p \in \mathbb{I}, x_0 \in \mathbb{I}$, and f does not contain envelope functions, being $\mathbb{I}$ the set of closed intervals of $\mathbb{R}$. These functions represent a dynamical systems family depending on p, x_0 and r. It is denoted as *semiqualitative model* and it is represented further on

$$\dot{x} = f(x, p), \qquad x(t_0) = x_0, \qquad \Phi_0(p, x_0) \tag{4}$$

where p and r have been joined in an unique parameters vector p.

3 Semiqualitative analysis

Qualitative analysis of a dynamical system intends to analyze the *phase portrait* or *phase space* of the system. The *phase space* of the dynamical system is constituted by the variables of state x, and the *extended phase space* by variables and parameters x, p. The phase portrait is formed by the projection of the trajectories of the dynamical system in the extended phase space. The phase portrait is interpreted as a correspondence between the differential equations and the vector field. In this paper semiqualitative systems that they are stable structurally are studied. In them, little perturbations keep their qualitative behaviors.

The first step of semiqualitative analysis of a dynamical system (4) is the determination of the *equilibrium regions*. They are defined by the constraints

$$Equilibrium(x,p) \equiv \{ f(x,p) = 0, \tag{5}$$

The study of solutions of (5) let us know the structure of the phase portrait. Each stable equilibrium region is an attractor region.

The *stability* of each equilibrium region is related to the real part of the eigenvalues of the Jacobian of the system. It has been demonstrated in the bibliography that in the stable fixed points the real part of the eigenvalues is negative. In order to apply the stability criteria, it is necessary to construct the following determinants. They are formed with the coefficients of the characteristic polynomial P_n of the Jacobian matrix A of the dynamical system. The Jacobian matrix of (4) is $A = D_x f(x,p)$, and P_n is defined as

$$P_n(\lambda) = det(A - \lambda I) = a_0\lambda^n + a_1\lambda^{n-1} + \ldots + a_{n-1}\lambda + a_n \tag{6}$$

In order to determine the stability conditions, the matrices are defined

$$g_i = det \begin{pmatrix} a_1 & a_3 & a_5 & \ldots & a_{2i-1} \\ a_0 & a_2 & a_4 & \ldots & a_{2i-2} \\ \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & 0 & a_i \end{pmatrix} \text{ being } i = 1, \ldots, n \tag{7}$$

The elements a_k of g_i are the coefficients of P_n for $k > n$, and 0 for $k \leq n$. Both a_k and g_i are symbolic expressions dependent on x, p.

We can apply two stability criteria. First is the *Routh-Hourwitz criterion.* For this criterion the predicate *Stable_Pol* is defined as

$$Stable\_Pol(P_n(\lambda)) \equiv \{ g_1 > 0, \ldots, g_n > 0 \tag{8}$$

Second is the *Lineard-Chipard criterion.* It defines the predicate *Stable_Pol* as

$$Stable\_Pol(P_n(\lambda)) \equiv \begin{cases} a_1 > 0, \ldots, a_n > 0, \\ g_{n-1} > 0, g_{n-3} > 0, \ldots \end{cases} \tag{9}$$

Therefore the constraints that define the stable equilibrium regions are

$$Stable(x,p) \equiv \begin{cases} Equilibrium(x,p), \quad A = D_x f(x,p), \\ P_n = P_C(A), \quad Stable\_Pol(P_n(\lambda)) \end{cases} \tag{10}$$

where D_x stands for the Jacobian, and P_C stands for the set of constraint of characteristic polynomial. If constraints (10) are satisfied by an *equilibrium region*, it is stable. Otherwise it is not stable.

The study of the *bifurcations points* of a system intends to divide the parameters space in regions. The system has the same number and type of attractors in these regions. The frontiers of these regions are formed by bifurcation points. An attractor appears, disappears or changes of type, when we cross a determined frontier.

The most elemental classification of bifurcation points distinguishes them into statics and dynamics. The *statics bifurcation points* are the simplest. They appear in those points where the number of attractors points varies. The determinant of the Jacobian matrix is annuled in them, that is, the characteristic polynomial has a null root.

The *dynamic bifurcation points* involve *limit cycles* or *strange attractors*. We study the *Hopf bifurcation*, where an attractor point is converted into a limit cycle or vice versa. In these bifurcation points the characteristic polynomial of the Jacobian matrix has a pair of roots with real part equal to zero.

$$Sta\_Bif(x,p) \equiv \begin{cases} Equilibrium(x,p), \\ A = D_x f(x,p), \\ P_n = P_C(A), \\ P_n = \lambda\, Q_{n-1}, \\ Stable\_Pol(Q_{n-1}) \end{cases} \quad Din\_Bif(x,p) \equiv \begin{cases} Equilibrium(x,p), \\ A = D_x f(x,p), \\ P_n = P_C(A), \quad (11) \\ P_n = (\lambda^2 + w^2)\, Q_{n-2}, \\ Stable\_Pol(Q_{n-2}) \end{cases}$$

It is interesting to notice that all the predicates defined between (5) and (11) are formuled as interval constraint satisfaction problems. They are solved by adequate consistency techniques [5].

4 A biometallurgical system

4.1 Description and determination of the model

For a long time, it has been observed natural transformations of the sulphur and iron compounds. They are originated from the dissolution of minerals. Presence of iron-oxiding bacteria in mining areas and their acid drainages has been reported repeatedly.

Thiobacillus Ferrooxidans is considered to be the most important organism for the bacterial leaching of minerals. In indirect leaching the bacteria generate ferric iron by oxidizing soluble ferrous iron. The global reaction is

$$2Fe^{2+} + 1/2O_2 + 2H^+ \Longrightarrow 2Fe^{3+} + H_2O \qquad (11)$$

This method for production of acidified ferric solutions is used because ferric iron in turn oxidizes other metals in mineral, transforming them in the soluble form, and because it avoids ecological contamination problem of industrial extraction of metals from the rocks.

If the equation of Michaelis-Mention is applied to the reaction (11), then oxidation rate V is calculated as follows

$$V = V_{max}\frac{[S]}{k_m + [S]} \tag{12}$$

where V_{max} is maximum rate that it can be reached by increasing in the substrate concentration, $[S]$ is substrate concentration, and k_m is Michaelis constant. This constant stands for the concentration which the reaction rate is half of the maximum rate. This equation has two problems: the concentration bacterian is not constant and it cannot be applied to the bacterian growth because it is exponencial. Due to the complexity of the factors that take part in the bacteria oxidation of Fe(II) in Rotating Biological Contactors (*RBC*), as shown in figure (1). It has not been possible to determine a general mathematical model for this process. However, it has been proved that the bioxidation reaction continues a kinetic of first order with respect to the substrate concentration. In the experimentation there are two interconnected RBC. In them it is introduced a flow Q with an ferrous iron concentration.

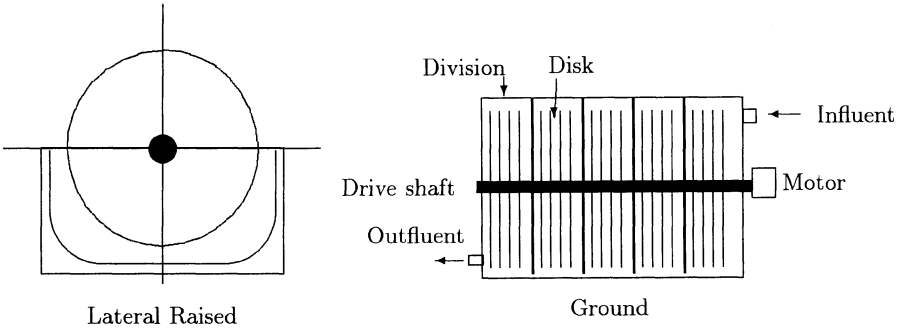


Fig. 1. A Rotating Biological Contactors (*RBC*)

The equations of the model of this dynamical system are

$$\begin{cases} \frac{dx_1}{dt} = \left(p_1\, p_2 - p_3\, p_4 \frac{x_1}{x_1+p_5} - p_1 x_1\right) p_6 \\ \frac{dx_2}{dt} = \left(p_1\, x_1 - p_7\, p_8\, \frac{x_2}{x_2+p_9} - p_1 x_2\right) p_{10} \end{cases} \tag{13}$$

4.2 Experimental data

According to the experimental results [3] it has been determined the quasi-equilibrium points for the system. They have been obtained studying different influent flows p_1 and values of iron concentration in such flows p_2.

According to the data supplied by the experts p_5 is similar to p_9 and their order of absolute magnitude is *moderately positive*, p_7 is *very positive*, and p_3 is *slightly greater* that p_7. Therefore, if it is associated the corresponding intervals to the previously expressed qualitative operators, it is obtained

$$\begin{array}{l} p_1 = 0.61l/h, p_2 = 3.96g/l, \quad p_3 = [5.6, 5.8], p_4 = 0.74l, \qquad p_5 = [0.4, 0.5], \\ p_6 = 0.015, \quad p_7 = [5.3, 5.6], p_8 = 0.78l, \qquad p_9 = [0.4, 0.5], p_{10} = 0.01 \end{array}$$

Using these data and applying the exposed techniques, we carry out the semiqualitative analysis of these dynamical systems.

4.3 Semiqualitative analysis

The semiqualitative analysis of this system is carried out to study how to increase the capacity of production, when systems parameters are varied. The equilibrium regions of the system are determined solving the network of contraints

$$Equilibrium(x,p) \equiv \begin{cases} \left(p_1\, p_2 - p_3\, p_4 \frac{x_1}{x_1+p_5} - p_1 x_1\right) p_6 = 0, \\ \left(p_1\, x_1 - p_7\, p_8\, \frac{x_2}{x_2+p_9} - p_1 x_2\right) p_{10} = 0, \\ 0.4 \leq p_5 \leq 0.5,\, 0.4 \leq p_9 \leq 0.5,\, 5.6 \leq p_3 \leq 5.8, \\ 5.3 \leq p_7 \leq 5.6,\, p_1 = 0.61,\, p_2 = 3.96, \\ p_4 = 0.74,\, p_6 = 0.015,\, p_8 = 0.78,\, p_{10} = 0.01 \end{cases}$$

If it is applied interval arithmetic the results obtained are too wide. Nevertheless it is applied interval consistency techniques developed in [5] and we will obtain a narrowing equilibrium region

$$Equilibrium(x,p) = \{[0.397, 0.49], \times [0.0198, 0.034]\}$$

This solution includes all experimental results obtained from different experience data.

The Jacobian matrix of this model is

$$A = \begin{pmatrix} a_{11} \; a_{12} \\ a_{21} \; a_{22} \end{pmatrix} = \begin{pmatrix} -\frac{p_3 p_4 p_5 p_6}{(x_1+p_5)^2} - p_1 p_6 & 0 \\ p_1 p_{10} & -\frac{p_7 p_8 p_9 p_{10}}{(x_2+p_9)^2} - p_1 p_{10} \end{pmatrix}$$

The characteristic polynomial of A is

$$P_n(\lambda) = a_0 \lambda^2 + a_1 \lambda + a_2 = \lambda^2 + (-a_{11} - a_{22})\lambda + a_{11}a_{22} - a_{12}a_{21}$$

and according to the Lienard-Chipard criterion,

$$Stable\_Pol(P_n(\lambda)) \equiv \left\{ (-a_{11} - a_{22}) > 0, a_{11}a_{22} - a_{12}a_{21} > 0 \right.$$

Substituting p_i for their values and simplifying, the constraints that define the stability are

$$Stable(x,p) \equiv \begin{cases} Equilibrium(x,p), \\ -0.00915 + \frac{0.0111[5.6,5.8]x_1}{([0.4,0.5]+x_1)^2} - \frac{0.0111[0.4,0.5]}{[0.4,0.5]+x_1} > 0 \\ -0.00915 + \frac{0.0111[5.6,5.8]x_1}{([0.4,0.5]+x_1)^2} - \frac{0.0111[5.6,5.8]}{[0.4,0.5]+x_1} - 0.0061+ \\ \frac{0.078[5.3,5.6]x_2}{([0.4,0.5]+x_2)^2} - \frac{0.078[5.3,5.6]}{[0.4,0.5]+x_2} > 0 \end{cases}$$

These constraints are satisfied with the obtained equilibrium region and therefore it is conclude that the region is stable.

The constraints that define the bifurcations are

$$Sta\_Bif(x,p) \equiv \begin{cases} Equilibrium(x,p), \\ a_1 > 0,\ a_2 = 0 \end{cases} Din\_Bif(x,p) \equiv \begin{cases} Equilibrium(x,p), \\ a_1 = 0,\ a_2 > 0 \end{cases}$$

When it is applied constraint satisfaction techniques to these constraints, there are no solutions, and hence the system has no bifurcations.

5 Conclusions

This paper proposes a method to carry out automatically the semiqualitative analysis of dynamical systems by interval consistency techniques. Qualitative knowledge is represented by intervals, and they are qualitative operators and envelope functions.

It has been applied the proposed approach to systems appeared in the bibliography and the obtained results are quite similar to them. In this paper, it has been studied a real biometallurgic system. The achieved results have allowed to know how to increase the capacity of production.

In the future, we are going to apply the previous techniques to other real problems. We also want to extend the analysis process with the study of other types of attractors, dynamic bifurcations, and the incorporation of multiple scales of time, and delays.

References

1. Abelson H., *The bifurcation interpreter: a step towards the automatic analysis in dynamical systems.* Int. J. Computers Math. Applic. Vol 20, nª8 13-35, 1990.
2. Dvorak D.*Monitoring and diagnosis of continuous dynamic systems using semiquantitative simulation* Ph.D. Dissertation, University of Texas. Tech. Report AI92-170, 1992.
3. Garcia M.J., *La Oxidación biológica en continuo del ión ferroso en reactores de pelicula bacterian soportada.* Ph.D. Thesis. Universidad de Sevilla. 1992.
4. Gasca R.M., Ortega J.A., Toro M. *Representación y simulación de modelos integrando conocimiento cualitativo y cuantitativo.* In Proc. VII Conf. Asoc. Española para Inteligencia Artificial, 1997 (to appear).
5. Gasca R.M. *Razonamiento y simulación en sistemas que integran conocimiento cualitativo y cuantitativo* Ph.D. Thesis Universidad de Sevilla, 1998.
6. Kay H., Kuipers B., *Numerical behavior envelopes for qualitative models.* Proc. 11th National Conf. on Artificial Intelligence, 606-613, 1993.
7. Kay H. *Refining Imprecise models and their behaviors* Ph.D. Tesis, University of Texas, 1996.
8. Kuipers B.J., *Qualitative simulation*, Artifical Inteligence 29,289-338, 1986.
9. Sacks E., *Automatic qualitative analysis of dynamic systems using piecewise linear approximations*, Artificial Intelligence 41, 313-364, 1990.

10. Sacks E., *Automatic analysis of one-parameter planar ordinary differential equations by intelligent numeric simulation* Artificial Intelligence no48, 27-56, 1991.
11. Vescovi M., Farquhar A., Iwasaki Y., *Numerical interval simulation: combined qualitative and quantitative simulation to bound behaviors of non-monotonic systems* Proc. 14th Int. Joint Conf. on Artificial Intelligence, 1806-1812, 1994.
12. Yip K.M., *KAM: A system for intelligently guiding numerical experimentation by computer*, MIT Press, 1991.
13. Zhao F., *Extracting and representing qualitative behaviors of complex systems in phase space.* Artificial Intelligence no 69, 51-92, 1994.

Including Qualitative Knowledge in Semiqualitative Dynamical Systems

J. A. Ortega, R. M. Gasca, M. Toro

Universidad de Sevilla
Departamento de Lenguajes y Sistemas Informáticos,
Facultad de Informática y Estadística
Avda. Reina Mercedes s/n, Sevilla (España)
e-mail:{jaortega,gasca,mtoro}@lsi.us.es

Abstract. A new method to incorporate qualitative knowledge in semiqualitative systems is presented. In these systems qualitative knowledge may be expressed in their parameters, initial conditions and/or vector fields. The representation of qualitative knowledge is made by means of intervals, continuous qualitative functions and envelope functions.
A dynamical system is defined by differential equations with qualitative knowledge. This definition is transformed into a family of dynamical systems. In this paper the semiqualitative analysis is carried out by means of constraint satisfaction problems, using interval consistency techniques.

1 Introduction

In engineering and science, knowledge about dynamical systems may be represented in several ways. The models constructed for studying them are normally composed of qualitative as well as quantitative knowledge. Models which only incorporate quantitative knowledge (quantitative models) have been well studied. The techniques developed to analyse and simulate are well known, too.

On the other hand, a great variety of techniques have been studied for representation and manipulation of qualitative knowledge, such as algebra of signs, interval arithmetic, fuzzy sets, and order of magnitude reasoning.

The knowledge composed of quantitative and qualitative knowledge is known as semiqualitative. Real models contain quantitative, qualitative and semiqualitative knowledge, and all of them need to be considered when they are studied. Therefore, sometimes it is necessary to solve conflicts on the request of accuracy and flexibility. The models of dynamical systems should provide different levels of numerical abstraction for their elements. These levels may be purely qualitative descriptions [7], semiqualitative [1], [5], numerical based on intervals [14], quantitative and mixed of all levels [6].

In the sixties, the methodology of system dynamics was proposed. It incorporated qualitative knowledge to models by means of variables and quantitative functions suitably chosen. But, it is not until the eigthies when the interest for studying qualitative knowledge independently of its quantitative representation emerges. This interest appears around qualitative simulation [7], and qualitative analysis [11]. Mathematical concepts of quantitative analysis of dynamical systems have been applied into qualitative simulation and analysis (see [8]).

Qualitative methods for studying dynamical systems began at the end of the last century, by the French mathematician Henri Poincaré. The subsequent evolution of these works has originated the *qualitative theory of dynamical systems.* In [10] the techniques to carry out the analysis of the qualitative models were introduced, that is, the study of equilibrium regions, stability and bifurcation points.

In this paper, the qualitative knowledge is represented by means of real intervals, continuous qualitative functions and envelope functions. The intervals include all the real values where the qualitative label of such magnitude is found. A continuous qualitative function stands for a family of functions defined by means of landmarks. An envelope function stands for a family of functions included between a real superior function and an inferior one.

It is also presented A method to transform semiqualitative models into a constraint network is presented, as well. The interval constraint satisfaction problems are solved applying consistency techniques [3].

2 Semiqualitative models

A semiqualitative model is represented by

$$\Phi(\dot{x}, x, p), \qquad x(t_0) = x_0, \qquad \Phi_0(p, x_0) \tag{1}$$

being x the state variables of the system, p the parameters, $\dot{x}$ the variation of the state variables with the time, Φ_0 the constraints in the initial conditions, and Φ the constraints depending on $\dot{x}, x$ and p. They are expressed by means of the operators defined in the next section. They are composed of variables, constants, arithmetic operators, functions, qualitative functions and/or envelope functions. Therefore the equations (1) stand for a family of dynamical systems depending on p and x_0.

The integration of qualitative knowledge is made by adding constraints to the network. They are constraints combined with *'and'* and *'or'* operators. This representation will help us to obtain the behavior of the system if we apply an appropriate algorithm to solve the resulting constraint network (see [3]).

3 Representation of qualitative knowledge

We shall focus our attention in dynamical systems where there may be qualitative knowledge in their parameters, initial conditions and/or vector field. They constitute the semiqualitative differential equations of the system.

First, we need to take into account that the representation of the qualitative knowledge is carried out by means of operators. They have associated real intervals. This representation provides the following advantages: easy integration of qualitative and quantitative knowledge [2]; and it makes possible the definition of the range of qualitative variables and parameters of the system. This definition is provided by experts, and it allows for techniques developed on intervals analysis and constraint satisfaction problem to be used [4], [13] and [3].

3.1 Qualitative parameters and initial conditions

The qualitative representation of parameters and/or initial conditions of dynamical systems may be carried out by means of the *qualitative operators* U and B. They represent, respectively, the set of unary and binary qualitative operators. For example, $U = \{$*very negative, moderately negative, slightly negative, slightly positive, moderately positive, very positive* $\}$ and $B = \{$*much less than, moderately less than, slightly less than, much greater than,...*$\}$. Each operator $op \in U$ or $op \in B$ has associated a real interval I_{op}. This real interval denotes a quantity. It stands for those values where the magnitude has the qualitative label.

The *binary qualitative operators* are classified in two classes according to their types:

- Operators related to the difference. They can be *exactly equal to* $=$, *smaller or equal to* $\leq$, and *larger or equal to* $\geq$.
- Operators related to the quotient. They can be *much less than* $\ll$, *moderately less than* $- <$, *slightly less than* $\sim<$, *approximately equal to* $\approx$, *slightly greater than* $>\sim$, *moderately greater than* $> -$, and *much greater than* $\gg$

3.2 Envelope functions

These functions establish a possible range of values for its image for each given value. They represent the family of functions included between two defined function, a superior one $\overline{g} : I\!R \to I\!R$ and another inferior one $\underline{g} : I\!R \to I\!R$.

Let be $y = g(x)$ an envelope function (see figure 1.a). It is represented by means of

$$\langle \underline{g}(x), \overline{g}(x), I \rangle, \qquad \forall x \in I: \quad \underline{g}(x) \leq \overline{g}(x) \tag{2}$$

where I is the definition domain in the real line of g, and x is a variable.

3.3 Qualitative continuous functions

Let be $y = h(x)$ a qualitative continuous function. It represents a functional relationship with x as independent variable, and y as dependent variable.

$$y = h(x), \quad h \equiv \{P_1, s_1, P_2, ...s_{k-1}, P_k\} \text{ with } P_i = (d_i, e_i) \tag{3}$$

where each P_i is a point. It stands for an important qualitative landmark of h. Each P_i is represented by means of a pair (d_i, e_i) where d_i is the associated qualitative landmark to the variable x and e_i to y. Points are separated by the sign s_i of the derivative in the interval between a point and the following. The sign s_i is $+$ if the function is strictly monotonic increasing in that interval, $-$ if it is strictly monotonic decreasing, and 0 if it is constant. The definition of a function

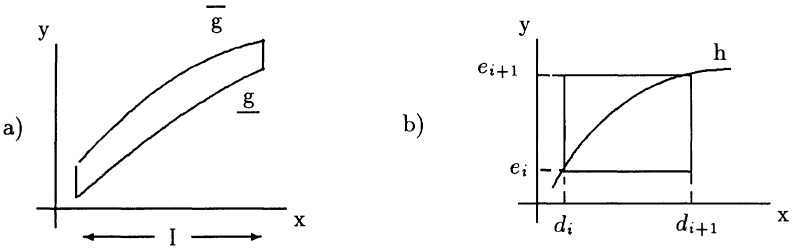


Fig. 1. Qualitative functions

is always completed with the landmarks which denote the cut points with the axes, and the points where the sign from the derivative changes (a maximum or a minimum of h). Qualitative interpretation of h (see figure 1.b) for each P_i is:

$$y - h(x) = 0 \equiv \begin{cases} x = d_i \Rightarrow y = e_i \\ d_i < x < d_{i+1} \Rightarrow \begin{cases} s_i = + \Rightarrow e_i < y < e_{i+1} \\ s_i = - \Rightarrow e_i > y > e_{i+1} \\ s_i = 0 \Rightarrow y = e_i \end{cases} \end{cases}$$

A special case of continuous function is that where the sign of all intervals is the same, that is, $s_1 = ... = s_{k-1} = s$. It is a strictly monotononic function. This function can be expressed in a short way by $h \equiv M^s\{P_1, P_2, ..., P_k\}$

4 From quantitative and qualitative knowledge to constraint networks

The qualitative knowledge is added by means of a set of constraints. They are combined with *'and'* and *'or'* operators. The constraints obtained from qualitative knowledge are:

- Qualitative parameters and initial conditions

For each operator, it is obtained a constraint according to its type. Let r be a new variable generated. I_u, I_b are the intervals associated to the unary and binary operators. Intervals I_u are stablished in accordance with [12], and intervals I_b with [9].

$\star$ Let u be an unary operator $u \in U$. Let e be an arithmetic expression. The resulting constraints are

$$u(e) \equiv \{e - r = 0, \quad r \in I_u$$

$\star$ Let b be a binary operator $b \in B$. Let op the binary operator, and let e_1, e_2 be two arithmetic expressions. If b is an operator related to the difference $(=, \leq, \geq)$ then the resulting constraints are

$$b(e_1, e_2) \equiv \{e_1 - e_2 \quad op \quad 0$$

and if b is an operator related to the quotient,

$$b(e_1, e_2) \equiv \{e_1 - e_2 * r = 0, \quad r \in I_b$$

- Envelope functions

For each envelope function $y = g(x)$ the constraint (4) is obtained

$$g(x) = \alpha \underline{g}(x) + (1-\alpha)\overline{g}(x) \qquad \alpha \in [0,1] \tag{4}$$

This constraint stands for a family of functions included between $\underline{g}$ and $\overline{g}$. It is interesting to notice that if $\alpha = 0 \Rightarrow g(x) = \overline{g}(x)$ and if $\alpha = 1 \Rightarrow g(x) = \underline{g}(x)$ and any other value of α in [0,1] stands for any included value between $\underline{g}(x)$ and $\overline{g}(x)$.

- Qualitative functions

For each qualitative function $y = h(x)$ defined as (3) is carried out the following :

⋆ For each landmark d_i or e_i appeared in the definition of h, it is added to the set of variables of the model a new variable with a domain $(-\infty, +\infty)$.

⋆ The following linear constraints due to the definition of h are added to the constraint network

$$y - h(x) = 0 \equiv \begin{cases} d_1 < d_2 < \ldots < d_k, e_1 \diamond e_2 \diamond \ldots \diamond e_{k-1} \diamond e_k, \\ ((x = d_1, y = e_1); \ldots; (x = d_k, y = e_k); \\ (d_1 < x < d_2, e_1 \diamond_1 y \diamond_1 e_2); \ldots; \\ (d_{k-1} < x < d_k, e_1 \diamond_{k-1} y \diamond_{k-1} e_k)) \end{cases} \tag{5}$$

where (5) is a set of linear constraints combined with *and* and *or* operators, respectively denoted by comma (,) and semicolon (;). The operator $\diamond$ is defined as

$$x \diamond_i y \equiv \begin{cases} x > y & \text{if } s_{i-1} = s_i = + \\ x < y & \text{if } s_{i-1} = s_i = - \\ x = y & \text{if } s_{i-1} = s_i = 0; s_{i-1} \neq s_i \end{cases}$$

The set of constraints obtained by the inclusion of qualitative knowledge to the model is formuled as an interval constraint satisfaction problem. As we have indicated, a constraint-based reasoning method by means of interval consistency techniques is applied (see [3]). The results obtained are a set of real intervals by the variables and constraints order among them.

5 An example

Let be two interconnected tanks (see figure 2). The semiqualitative model is

Variables: $V = \{x_1, x_2, r_1, r_2, s, p\}$
Landmarks : $L = \{a, b\}$
Functions: $h_1 \equiv M^+\{(0,0), a, b\}$
$g_1 \equiv < 2x, x, [0, \infty] >$
Constraints: $r_1 = h_1(s)$, $r_2 = g_1(x_2)$, $s = x_1 - x_2$
$\frac{dx_1}{dt} = p - r_1$, $\frac{dx_2}{dt} = r_1 - r_2$, positive medium($p$)

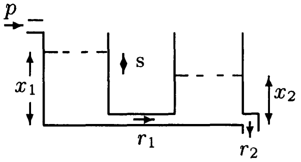


Fig. 2. The interconnected tanks system

The constraint network obtained applying the proposed techniques is

$$\begin{cases} V = \{x_1, x_2, r_1, r_2, s, p, a, b\} \\ 0 < a, \quad 0 < b, \\ (\,(s < 0, r_1 < 0); \quad (s = 0, r_1 = 0); \quad (0 < s < a, 0 < r_1 < b); \\ \ (s = a, r_1 = b); \quad (s > a, r_1 > b)\,), \\ r_2 = \alpha(x_2) + (1 - \alpha)2x_2, \quad \alpha \in [0, 1], \\ s = x_1 - x_2, \quad \frac{dx_1}{dt} = p - r_1, \quad \frac{dx_2}{dt} = r_1 - r_2, \quad p \in [3, 5] \end{cases} \tag{6}$$

The concepts introduced in [10] have been applied in order to carry out the analysis of this model. The constraint network that stands for the equilibrium regions of the model are replacing the expressions $dx_1/dt, dx_2/dt$ by zero in (6). This constraint satisfaction problem has an unique solution, hence there is an equilibrium region where it is satisfied that

$$x_2 \in I_{x_2}, \qquad x1 > x2 \tag{7}$$

being I_{x_2} a real interval. If it is applied interval arithmetic to (6), the intervals obtained for x_1, x_2 are too wide. The equilibrium region is $[0., \infty] \times [0., \infty]$. In order to narrow this solution it is applied the consistency techniques [3], and then it is obtained for x_2 the interval $[1.06487, 5.0]$. The intervals for x_1, s are positives, and using the constraint $s = x_1 - x_2$, then results (7) are concluded. Therefore the solution for I_{x_2} is closed to the real solution $[1.5, 5.0]$.

The results (7) may be interpreted as: the system has an unique equilibrium where the height of the first tank is higher that the second one. The height of second tank is in the real interval $[1.5, 5.]$.

In a similar way, it is obtained the network that stands for the stability of such region. This network is also satisfied, hence it is an stable equilibrium region. The constraint network that define the bifurcations points are not satisfied. Therefore it is concluded that there are no bifurcations.

6 Conclusions

This paper provides a method for including qualitative knowledge in semiqualitative dynamical systems. Qualitative knowledge is represented by means of intervals, continuous qualitative functions and envelope functions. This knowledge helps us to make analysis of that kind of systems.

We have applied the method proposed to several examples. The obtained results have been satisfactory. The technique presented is appropriate for predictive problems in industrial processes where there is qualitative information of their components. At the moment, we are applying the method to study a real biometallurgic system.

References

1. Dvorak D.*Monitoring and diagnosis of continuous dynamic systems using semiquantitative simulation* Ph.D. Dissertation, University of Texas. Tech. Report AI92-170 (1992)
2. Gasca R.M., Toro M., Ortega J.A., *Propagación de restricciones integrando conocimiento cualitativo y cuantitativo* Boletín de AEPIA núm 6 (1996) 23-30
3. Gasca R.M. *Razonamiento y Simulación en Sistemas que integran conocimiento cualitativo y cuantitativo* Ph.D. Thesis, Universidad de Sevilla (1998)
4. Hyvönen E.*Constraint reasoning based on interval arithmetic: the tolerance propagation* Artificial Intelligence no. 58 (1992) 71-112
5. Kay H., Kuipers B., *Numerical behavior envelopes for qualitative models.* Proc. 11th National Conf. on Artificial Intelligence (1993) 606-613
6. Kay H. *Refining imprecise models and their behaviors* Ph.D. Tesis, University of Texas (1996)
7. Kuipers B.J., *Qualitative simulation*, Artifical Inteligence 29, (1986) 289-338
8. Lee W.W., Kuipers B.J., *A qualitative method to construct phase portraits*, Proc. 11th National Conf. on Artificial Intelligence (1993) 614-619
9. Mavrovouniotis M.L., Stephanopoulos G. *Formal order-of-magnitude reasoning* Process Engineering Computer Chemical Engineering no. 12 (1988) 867-880
10. Ortega J.A., Gasca R.M., Toro M., *Automatización del análisis y simulación de modelos cualitativos mediante restricciones* Proc. VII Conf. Asoc. Española para Inteligencia Artificial, (1997) 85-94
11. Sacks E., *Automatic qualitative analysis of dynamic systems using piecewise linear approximations*, Artificial Intelligence 41 (1990) 313-364
12. Travé-Massuyès L., and Piera N.,*The order of magnitude models as qualitative algebras* Proc. of 11th Int. Joint Conf. on Artificial Intelligence, Detroit MI (1989) 1261-1266
13. VanHentenryck P., McAllester D., Kapur D., *Solving polynomial systems using a branch and prune algorithm.* Tech. report CS-95-01 University of Providence (1995)
14. Vescovi M., Farquhar A., Iwasaki Y., *Numerical interval simulation: combined qualitative and quantitative simulation to bound behaviors of non-monotonic systems*, Proc. 14th Int. Joint Conf. on AI (1994) 1806-1812

A Tool to Obtain a Hierarchical Qualitative Rules from Quantitative Data

Jesús Aguilar, José Riquelme y Miguel Toro.

Departamento de Lenguajes y Sistemas Informáticos.
Facultad de Informática y Estadística. Universidad de Sevilla.
Avda. Reina Mercedes s/n. 41012 Sevilla.
e-mail: {aguilar, riquelme, mtoro}@lsi.us.es

Abstract. A tool to obtain a classifier system from labelled databases is presented. The result is a hierarchical set of rules to divide the space in n-orthohedrons. This hierarchy means that obtained rules must be applied in specific order, that is, an example will be classify by i-rule only if it isn't matched the conditions of the i-1 preceding rules. It is used a genetic algorithm with real codification as searching method. Logically, computation time will be greater than other systems like C4.5, but it will provide flexibility to the user because it is always possible to produce rules set with 0% of error rate and, from here, to relax the error rate for having less rules. Afterwards, a qualitative approach is made to obtain a linguistic rule set. Finally, several results are summarized in section 4.

1 Introduction

C4.5 [1] is one of the most extended programs for supervised learning by several qualities: easy use, low computation time, clear interpretation of the results and acceptable error rate. Its algorithm learns decision trees by constructing them top-down, beginning with the question "which attribute should be tested at the root of the tree?" The best attribute is selected by using a statistical test to determine how well it alone classifies the training examples. A descendant of the root node is then created for each possible value of this attribute, and the training examples are sorted to the appropriate descendant node. This forms a greedy search for an acceptable decision tree, in which the algorithm never backtracks to reconsider earlier choices. Therefore, it is susceptible to the usual risks of hill-climbing search without backtracking: converging to locally optimal solutions that are not globally optimal.

Genetic algorithms employ a randomized search method to seed a maximally fit hypothesis. This search is quite different from other learning methods. The genetic algorithm search can move much more abruptly, replacing a parent hypothesis by an offspring less likely to fall into the same kind of local minima that can happen with other methods, like C4.5.

Some training files are very difficult to classify. Below, it is presented a file with two parameters and two classes: white or black circles. The most immediate classification of these data is to select three rectangles of black circles and the remainder of white circles, that is, four rules. However, C4.5 produces a set of thirteen rules, which are presented in the figure on the right.

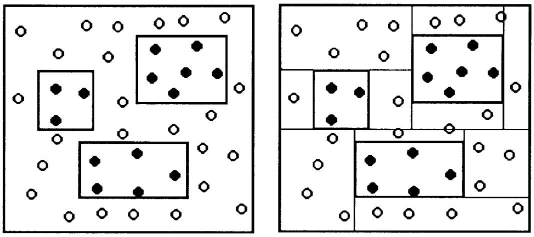


Fig. 1. Hierarchical rules vs C4.5

Sometimes it can be useful that user sets a maximum error rate for having a rules set with the least number of rules, according to the initial error rate. Or vice versa, this is, user establishes the number of rules that he wants and the system produces a rules set with the minimum error rate for that number of rules.

Our approach for solving the disadvantages of C4.5 is to produce as solution a hierarchic rules set. Here, rules must be applied in specific order. With this organization, the number of rules is substantially reduced. The shape of these rules would be:

```
If conditions then A-class
else  If conditions then B-class
      else If .................................
                       else Z-class.
```

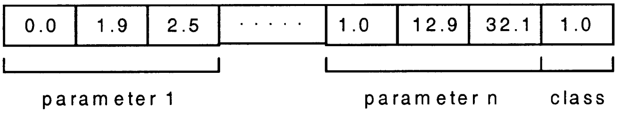
The objective is to design a system able to obtain a decision rules set from a labelled database. We propose a genetic algorithm [2] with real codification for the individuals [3]. GABIL [4] and GIL [5] are the most known tools. Both have binary codification but these can not work well with continuous spaces.

Finally, in [6] we presented a first version of this work, which produces a rules set without errors by using genetic algorithms with binary codification. Here, in addition to use real codification, it is possible to relax the error rate what we wants for having less rules and for making more comprehensible the linguistic model that rules set offers.

2 Description of the system

2.1 The environment and its codification

Information of the environment come from a data file, where each example has a class and a number of parameters. The number of parameters is fixed but undefined, like the number of classes. The genetic algorithm uses real codification, that is, an individual is formed by a n-tuple of real, including class, that it takes real values beginning with 1 and adding a unit to the successive. We consider that an example is a member of k-class if it satisfies its conditions, which are defined with upper and lower boundaries for each parameter. So, the representation of an individual takes the following form:



Each parameter is defined by,

- A value belonging to the set {0, 1, 2}, that identifies the type of operation:
  - 0 means "if pl ≤ 1.9..."
  - 1 means "if pl ≥2.5..."
  - 2 means "if 1.9 ≤ pl ≤ 2.5..."
- The two following values are the limits for each parameter. If the operator is ≤ only makes sense the first value; if the operator is ≥, the second value is used; and, in the third case the two values indicates lower and upper boundaries.
- The last value identifies the class. It will exists as many possible values as classes, that is to say, if there are five classes, the value will belong to the set {0,1,2,3,4}.

2.2 The algorithm

The algorithm is:

```
initialize
repeat
  evolution
  select rule
  adapt environment
until stop condition
```

The *initialize* module calculates the number of data of the training file, the number of parameters per datum and the number of classes. It also calculates the maximum and minimum values for each parameter. The *evolution* module is a classic evolutionary algorithm with real codification which is seeking a rule to minimize the fitness function. The *select rule* module chooses the best individual of the evolutionary process, transforming it into a rule. This rule is used in *adapt environment* to eliminate data from the training file that fulfill its premise (though they do not satisfy the conditions). In this way, the training file is reduced for the following iteration. *Stop condition* can be reached when the training file have been totally covered or when a number of rules have been obtained, or when it has been reached a precision. It is a typical sequential covering algorithm [7].

2.3 Initial population

The initial population is randomly generated. First, two values are randomly obtained between the boundaries of each parameter. An valid individual is formed by the values that belongs to the set of intervals before obtained. However, the examples that are near to the limits are hard to cover during the evolutive process. For solving it, the search space is increased. For thus, the lower bound is decreased a 5% of the range, and the upper bound is increased in the same value. To the end of the process, if the bound of a interval is out of range, this is replaced by the original limit of the database search space.

2.4 Genetic operations

The main feature of the evolution module presents is elitism, in the sense that the best of every generation is replicated to the next one and a set of children are obtained from copies of the parents, selecting it randomly but depending on their fitness values. The rest of individuals are formed through crossovers and after applying a mutation, depending on a probability of mutation.

We used two kind of crossover which are alternatively applied depending on a probability. On the one hand, it has been preserved the "n-point random crossover", that takes advantage of parameter values which can be good for some individual. The n-point random crossover chooses randomly a number k of crosspoints and the offspring (two individuals) inherits pieces between the crosspoints. The usefulness of this crossover is justified since can exist individual whose n-1 parameters may have good values but not the remaining parameters; then, they are crossed these values with other individuals before rejecting the defective individual. When n-point random crossover is applied with real codification do not generate new values for the parameters, on the contrary that binary codification. On the other hand, it has been used a real crossover specifically designed. The real crossover uses three weighted types depending on the significance that they could have in relation to the solution:

- intermediate segmented crossover: it obtains a random value belonging to the defined segment by two values which occupies the same location in both selected individuals (it is applied approximately the 40 % of the times).
- segmented crossover forced to the minimal (maximal): it obtain a random value belonging to the defined segment by the lower (upper) bound of the range and the smaller (greater) of the two values which occupies the same location in both selected individuals (it is applied approximately the 40 % of the times each one).

Mutation makes to grow the region covered by the individual so that it could take in more examples of the database. Two kinds of mutation have been used:

- incremental mutation: if it is right value of the parameter, the value is incremented a small quantity (1 % of the range); and if it is the left value of the parameter, it is decreased the same percentage.
- mutation forced to the boundary: one of the boundaries is mutated depending on whether it is the right value or the left value of the parameter range. This will make with very small probability (5% of the mutations).

2.5 Fitness function

The evolutionary algorithm minimizes the fitness function *f* for each individual. It is given by

$$f(i) = \frac{1}{1 + ND - g(i)} \tag{1}$$

where ND is the cardinality of the training file and *g* is a penalty function. This function presents two relevant features: first, the difference between class and data errors; and second, the use of a new factor called coverage. Class errors are produced when an example is covered by a rule even though the classes aren't the same. Class errors are produced when an example isn't covered by a rule, independently of the class.

Due to the different influence of that errors types in the learning task, it is necessary to introduce two penalty factors, one for each error type, called class error and data error penalty, respectively. The rule coverage is the side of a n-dimensional hypercube which volume is equivalent to the volume of the covered n-dimensional region by the rule. This new operator, applied to the population, allows to increase or to reduce the region of a rule without loosing matched examples. In this way, every rule can best adapt to the space by reducing its volume or, on the contrary, it can quickly expand for finding more examples. Particularly, our approach rewards to individuals that covers more space with same number of matched examples.

2.6. Relaxing coefficient

Databases uses as training files have not areas clearly differentiated in n-orthohedrons, for that, to obtain a rules system totally coherent involves a high number of rules. We show in previous paper a system (that we call COGITO) capable of producing a rules set exempt from error rate; however sometimes, it is interesting to reduce the number of rules for having a rules set that it can be used like a comprehensible linguistic model. When databases present a distribution of examples very hard to classify, then it is advisable to use a relaxing coefficient. Many times, we are more interested to understand the structure of databases than error rate. In this way, it could be better a system with less cardinality (despite some errors) than too many rules (with 0% of error rate). Then, it could be interesting to introduce the relaxing coefficient for understanding the behaviour of databases by decreasing the number of rules. Relaxing coefficient indicates what percentage of examples inside of a indicated region can have different class to the individual. For example, if we allow a relaxing coefficient of 10%, it means that an individual covering 83 examples of the class 'A' could make 8 errors as maximum. Relaxing coefficient behaves like the upper bound of the error with respect to the training file.

The present version of COGITO allows to limit the number of rules that we want to obtain and it can fix a maximum for the error rate. Consequently, two kinds of questions we could tackle: users can fix a maximum error rate and system produces a rules set minimizing the number of these; or users fix a maximum number of rules and the system produces a classifier minimizing the error rate. Results are very satisfactory in both cases since user directly controls the error rate. For example, if we fix the relaxing coefficient (R=20) and it obtains a error rate equal to 16 with 23 rules, the user can decide: whether to increase R if the number of rules is too high, or to decrease R if the error rate is too high. In this way, users can get the rules sets more beneficial for its application.

3 Qualitative Rules

3.1 Linguistic terms definition

The technique to obtain a qualitative information of a spatial arrangement, expressed in the previous paragraphs can be in a way simple converted automatically in a model based on linguistic terms.

To express through a linguistic term a value range of a variable, is a relatively easy task. Only it must take into account two considerations: the first one is that the number of terms that are defined must be enough to attempt to cover most of linguistic nuances of the possible value ranges. The second is that the number should not be excessive so as to complicate the understanding of the model. To choose the number of terms together with the nomenclature of these is, then, the only difficulty of this approximation. In this case the linguistic model would follow the following grammar:

```
<model>          ::= <list_rules>
<list_rules>     ::= <list_rules> <rule>
<rule>           ::=IF <list_premises> THEN <conclusion>
                    | <rule>
<list_premises> ::= <list_premises> AND <premise>
                    | <premise>
<premise>        ::= PARAMETER IS <list_terms>
<list_terms>     ::= <list_terms> OR <TERM> | <TERM>
                    | not <TERM>
<conclusion>     ::= <CLASS>
```

For the assignment between a set of values and what in the grammar has been expressed as <TERM> will have to be followed then steps:

1) The parameter *p* is supposed to have a range of values that remains defined by the minimal and maximum values of that parameter in the database. Those values will be *m* and *M*, respectively.

2) The interval of values is supposed to be divided into *L* linguistic terms, that may have an equivalence in *L* ranges of equal length values for the parameter *p*.

3) The central values of those ranges would come to consider the values succession:

$$m+\frac{M-m}{2L},\ m+\frac{3(M-m)}{2L},\ \dots,\ m+\frac{(2L-1)(M-m)}{2L} \qquad (2)$$

4) From these central values each range would have a value of $\leq(M\text{-}m)/2L$, that is, the first term would come defined by the range $[m,m+(M\text{-}m)/L]$, the second term would give value to the numbers of the range $[m+(M\text{-}m)/L,\ m+2(M\text{-}m)/L]$, and so on until $[m+(L\text{-}1)(M\text{-}m)/L,M]$.

3.2 Linguistic terms assignment to an interval

Let the interval $[c,C]$ that have been determined by COGITO for a given parameter. For the assignment of a rule to each interval the following methodology is proposed:

Notation:

C : if it exists superior bound.	v_i $1 \leq i \leq L$: central value of the ith term.
c : if it exists inferior bound.	t_i $1 \leq i \leq L$: ith linguistic term.
L : number of linguistic terms.	*TERM*: linguistic term assigned.

Method:

If $\exists\ c$, let r with $2 \leq r \leq L$ / $v_{r-1} < c$ & $v_r > c$ else $r=1$.
If $\exists\ C$, let s with $1 \leq s \leq L\text{-}1$ / $v_s < C$ & $v_{s+1} > C$
else $s=L$

TERM is calculated through the equation:

$$TERM = \bigcup_{i=r}^{s} t_i \tag{3}$$

where the union symbol indicates the conjunction of the terms by the logical operator or. If *TERM* is formed by the union of L-1 linguistic term, it is substituted for the expression *not TERM'* where *TERM'* is the linguistic term that is not in *TERM*. Finally, if a parameter have an interval that coincide with the range of parameter, then the correspondent premise is unnecessary.

4. Application

First, it has been applied to two files specifically created to demonstrate the weakness of C4.5 to find good regions in same cases. Afterwards, it has been applied to the standard databases for learning and classification of the UCI Repository [8]. We have compared results with C4.5, being presented in both cases the number of rules and error rate for every training file.

4.1 Bricks and boxes

The figure below shows a database (that we have called bricks) with two parameters and three labels, so that if the first rule collects the information of the central square, five hierarchic rules are sufficient to cover the whole database (left). However, C4.5 would need eight rules at least (right).

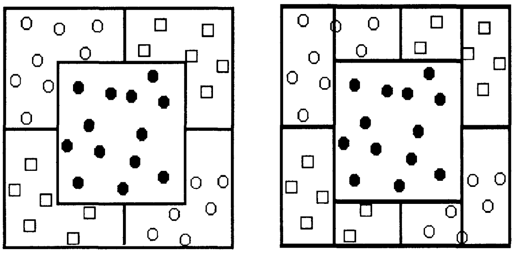


Fig. 2. Bricks database

Next figure shows a database (that we have called boxes), yet more extreme, since white or black circles are found distributed in squares one inside of the other. A hierarchical rules set only needs five rules while C4.5 produces not less than seventeen rules.

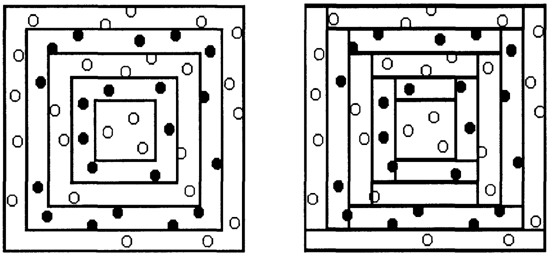


Fig. 3. Boxes database

Results, for both C4.5 and COGITO are shown in the table 1.

DATABASE	C4.5		COGITO	
	ER	NR	ER	NR
BRICKS	0	9	0	5
BOXES	0.8	27	0	5

Table 1. ER means error rate and NR means number of rules

This new version of COGITO improves the execution time on 25%. It can be asserted that a classifier system based on genetic algorithm with real codification is more powerful than other based on binary codification, since for a number of generations and a size of population fixed, the results are better in the first one than in the second one.

4.2 Databases of the UCI Repository

Below, they are presented the obtained results which are compared to those that C4.5 offers. In the two cases are indicated the number of rules and the error rate.

DATABASES	C4.5		COGITO	
	ER	NR	ER	NR
IRIS	1.3	6	0	6
WINES	1.1	5	0	3
PIMA	3.6 8.5	115 63	11.46	13
BUPA	6.1 9.6	55 35	13.91	7
BREAST CANCER	0.7	23	2.64	2

Table 2. Results for UCI Databases

It is notorious the difference between results offered by C4.5 and COGITO, above all, with respect to the number of rules. Though in the last three cases the error rate of C4.5 is lower, the number of rules is much greater, which makes complex the linguistic model. Furthermore, with all certainty, COGITO will obtain lower error rate than C4.5 for the same number of rules. Wine and Breast Cancer databases are surprising since they needs only one rule for each class -in the first case with unsurpassable efficiency-, what demonstrates the quality of the provided solutions.

In the following table it can be appreciated the range of error rate varying the number of rules for Pima database.

PIMA	5 R	6 R	7 R	8 R	11 R	12 R	13 R
TC	18.88	18.49	16.02	14.45	13.02	12.11	11.46

Table 3. Error rate for Pima Database with several NR

The executions have been accomplished in SUN SPARC 1000 E and, logically, execution time depends on the number of data of the training file and on the number of rules sought. For 200 individuals and 200 generations, to find the thirteen rules of Pima can take about 15 minutes, while the two rules of Breast Cancer about two minutes.

It could be a disadvantage the diversity of solutions since COGITO depends on probabilistic search. The tables shows the better of the possible executions, however, after realizing 20 executions to find 13 rules for Pima, the error rate range of found solutions was from 11.5 to 12.9, what from our point of view is not a relevant factor.

Below, it has shown two rule systems, Bupa and Breast Cancer, respectively, and number of goals/errors of each rule with respect to the training file.

Database BUPA:

Quantitative rules:

If p1≤97.0 y 44.0≤p2 y p3≤70.0 y 22.0≤p4 y p5≤297 y p6≤16.0 then class=A (79/15)

Else If p1≤103.0 y 19.5≤p2≤79.0 y p3≤20.0 y 5≤p4 y 7.0≤p5 y 0.0≤p6 then class=A (49/8)

Else If 87.0≤p1≤100.0 y 59.0≤p2 y 15.1≤p3 y p4≤47.8 y p5≤297 y 0.0≤p6 then class=B (81/18)

Else If p1≤94.0 y 23.0≤p2 y 15.1≤p3 y 14.0≤p4≤21.0 y p5≤30.0 y p6≤20.0 then class=B (18/1)

Else If 78.0≤p1 y 35.0≤p2≤119.0 y 11.0≤p3≤33.0 y 5.0≤p4 y 9.0≤p5 y p6≤20.0 then class=A (36/2)

Else If p1≤103.0 y p2≤134.0 y 4.0≤p3 y p4≤55.0 y p5≤48.0 y 0.0≤p6 then class=B (18/1)

Else If p1≤103.0 y 23≤p2 y p3≤155.0 y p4≤82.0 y p5≤297.0 y 0.0≤p6 then class=A (16/3)

Qualitative rules:

If p1 is not big and p2 is not small and p3 is small or medium and p4 is not small and p6 is not big then A

Else If p1 is small or medium and p2 is small or medium and p3 is small then A

Else If p1 is big and p2 is medium or big and p3 is not small and p4 is small or medium then B

Else If p1 is not big and p3 is not small and p4 is small and p5 is small then B

Else If p1 is small and p2 is medium and p3 is small then A

Else If p4 is small or medium and p5 is small then B

Else A

Database CANCER:

Quantitative rules:

If p1≤7.0 y p2≤6.0 y 1.0≤p3 y 1.0≤p4 y1.0≤p5 y p6≤6.3 y1.0≤p7 y p8≤8.3 y 1.0≤p9 then class=A (432/16)

Else If 1.0≤p1 y 1.0≤p2 y 1.0≤p3 y 1.0≤p4 y 1.0≤p5 y 1.0≤p6 y 1.0≤p7 y 1.0≤p8 y 1.0≤p9 then class=B (233/12)

Qualitative rules:

If p1 is not big and p2 is small or medium and p6 is small or medium and p8 is not big then A Else B

5 Conclusions

A supervised learning tool to classify databases in n-orthohedrons is presented. It produces a hierarchic rules set where conditions of each rule indicates the belonging of a example to an orthohedron. Relevant qualities of this system are two: important reduction of number of rules in comparison to C4.5 and flexibility to construct the classifier fixing boundaries to the number of rules or error rate through relaxing coefficient. The intervals defined by this system are easily translated to qualitative rules. The results as for bricks and boxes as UCI Repository databases has been very satisfactory.

References

1. Quinlan, J.R. C4.5: Programs for Machine Learning, Morgan Kaufmann Pub., 1993.
2. Goldberg, D.E. Genetic Algorithms in search, optimization and machine Learning. Addison-Wesley, 1991.
3. Michalewicz, Z. Genetic Algorithms + Data Structures = Evolution programs. Second Edition, Springer-Verlag, 1994.
4. De Jong,K.A. Using Genetic Algorithms for Concept Learning. Machine Learning, 93.
5. Janikow, C.Z. A Knowlege-Intensive Genetic Algorithm for Supervised Learning. Machine Learning, 93.
6. Aguilar, J., Riquelme, J. y Toro, M. COGITO: Un Sistema de Autoaprendizaje Basado en Algoritmos Genéticos. Actas de las III Jornadas de Informática. pp. 79-88., 1997.
7. Mitchell, T. Machine Learning. MacGraw-Hill, 1997.
8. Murphy, P. y Aha, D.W. UCI repository of machine learning databases. Dept. of Information and C.S. University of California, Irvine, 1994 .

Hybridization Techniques in Optical Emission Spectral Analysis

C.S. Ampratwum[1], P.D. Picton[1], A.A. Hopgood[2], A. Browne[1]

[1]Nene College, Northampton, NN2 6JD, UK
cecilia.ampratwum@nene.ac.uk
[2]The Open University, Faculty of Technology, Milton Keynes, MK7 6AA, UK

Abstract. The utilisation of formal artificial intelligence (AI) tools has been implemented to produce a hybrid system for optical emission spectral analysis that combines a multilayer perceptron neural network with rule-based system techniques. Even though optical emission spectroscopy is extensively used as an in-situ diagnostic for ionised gas plasmas in manufacturing processes, ways of interpreting the spectra without prior knowledge or expertise from the user's stand-point has encouraged the use of AI techniques to automate the interpretation process. The hybrid approach presented here combines a modified network architecture with a simple rule-base in order to produce explicit models of the identifiable chemical species.

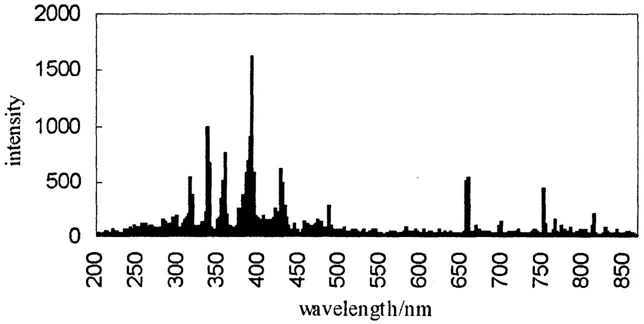
1 Introduction

Optical emission spectroscopy (OES) is typically implemented as a non-intrusive plasma process monitor and diagnostic instrument in semiconductor manufacturing [1], [2], [3]. For example, in semiconductor wafer etch processes a plasma is created that contains gaseous ions that bombard the surface of the material that is to be etched in order to fabricate an end product such as integrated circuits. To be of greatest use to the manufacturer, the OES system must be able to provide both dynamic information about the plasma process and specific information about the identity of trace impurities or etch by-products [4]. One of the goals behind this research is therefore to not just automate the identification of chemical species within a given plasma from its optical emission spectrum but to apply certain artificial intelligence (AI) techniques that will be able to identify species both accurately and efficiently. The aim is to produce a hybrid architecture that utilises some of the key advantages of certain AI techniques and can be used as a quick preliminary measurement tool in the automatic control of plasma processes that use OES.

2 Experimental and Discussion

To merely confirm what would be obvious to the eye of an expert spectroscopist when interpreting an optical emission spectrum (portrayal in Fig. 1) may be interesting but not particularly useful. The first step was to identify the chemical species that have prominent peaks in the optical emission spectrum of a mixed gas plasma. This is mainly done by recognising the unique patterns of optical emissions associated with most atomic and molecular species. Recognition is mainly a function of experience, and can be aided by comparison with reference spectra of common plasma species that have already been catalogued in spectral library databases [5], [6]. This process of identification can be implemented by applying the recognition capabilities of neural networks in order to automate that first species identification step that is part of the spectral interpretation process. Unfortunately, optical emission from many species overlap, and many emission lines of a species can vary or disappear, depending on how much and by what method energy is coupled into its chemical structure, therefore care must be taken when using spectral libraries to generate data. To overcome some of these problems associated with more complex spectral classifications particularly in mixed plasma spectra, a combination of a trained neural network with a rule-based system has been developed to enhance the accuracy and speed of species recognition.

Fig. 1. Optical Emission Spectrum of a 4-gas plasma



2.1 Neural Network analysis

2.1.1 Previous Work. Picton et al [7] showed that the architecture that worked for the spectral analysis of a mixed pattern (i.e. to be able to recognise the species within a mixed spectral pattern) was one that trained individual neurons on individual species using Kohonen learning. Each neuron is trained on normalised data since the presence of a peak within an optical emission spectral pattern is not reliant on the height of the peak but rather its peak location in the optical emission spectrum. Thus by sampling at specific wavelengths each neuron will receive a vector of unit length in order to be able to identify individual target species within a mixed spectral pattern. The network successfully recognised atomic argon (Ar) and atomic hydrogen (H) within the spectra from mixed argon and hydrogen plasmas. Advantages of this method are the quick training times and its adaptability to a specific plasma process and so it has been implemented into the preliminary stages of a diamond-like carbon deposition process which uses an optical emission spectrometer as a diagnostic integrated into a rule-based plasma control system [8].

2.1.2 Multilayer Perceptron. Since the multilayer perceptron (mlp) is still a plausible candidate for spectral analysis due to the cited references [6], [9], [10] that have applied its feedforward backpropagation network in quite successful interpretations of spectra such as infra-red (IR), nuclear magnetic resonance (NMR), mass, ultra-violet (UV) and ion mass spectrometry (IMS) to name but a few, apparently very little has been documented in applying the mlp to OES in particular. Therefore, three sets of experimentation on optical emission spectral analysis have been implemented to address this issue and identify the clear advantages as well as the shortcomings of the mlp architecture.

Experiment 1. The neural network implemented is a feed forward network based on the multilayer perceptron with no hidden layer of neurons. Thus it consists of an input layer and output layer only and employs the error backpropagation algorithm [10] in learning. The input layer receives as inputs normalised intensity values sampled at a set of wavelength points for a given optical emission spectrum. This procedure immediately reduces the dimensionality (dimensional space) of the OES spectrum resulting in rapid training times. The preprocessing of the data by normalisation to reduce the dimensionality is an inherent form of feature extraction. The values of these features are the inputs to the network. Selection of the most appropriate feature is very important as it has a direct bearing on the performance of the complete system, and so for this case the feature selection extracts the set of unique peak patterns (at wavelength bands) for a particular species. The same set of optical emission spectra from Picton et al's work [7] were used in this first experiment. The architecture adopted was a two-species detector (i.e. Ar and H detector) and was trained on the spectra from seven argon plasmas and seven hydrogen plasmas. Spectra from five mixed argon/hydrogen plasmas were then tested on the trained neural network and the results are shown in Table 1.

Table 1. Results from experiment 1 - species identification

files	Ar 1	Ar 2	Ar 3	Ar 4	Ar 5	Ar 6	Ar 7	(argon
Ar	0.96	0.97	0.97	0.97	0.97	0.97	0.97	plasmas)
H	***0.54***	***0.92***	***0.91***	***0.95***	***0.96***	***0.94***	***0.96***	
files	H 1	H 2	H 3	H 4	H 5	H 6	H 7	(hydrogen
Ar	0.29	0.21	0.23	0.27	0.24	0.18	0.19	plasmas)
H	0.96	0.97	0.97	0.96	0.96	0.96	0.96	
files	Ar/H 1	Ar/H 2	Ar/H 3	Ar/H 4	Ar/H 5	(mixed Ar/H2 plasmas)		
Ar	0.97	0.97	0.97	0.96	0.93			
H	0.94	0.94	0.95	0.94	0.90			

The results show that this architecture is successful in recognising the atomic argon, Ar, and atomic hydrogen, H, species in a mixed spectral pattern of argon and hydrogen. Also, it determined the presence of hydrogen (values highlighted in bold italics) in the spectra from argon plasmas. A value of *0* or *near 0* represents '*Species Absent*', *1* or *near 1* represents '*Species Present*'; the *threshold* is *0.5*, so a *value less than 0.5* is '*Species Absent*' and *greater than 0.5* is '*Species Present*'.

These results reflect the ability of a modified multilayer perceptron neural network architecture to identify two individual spectral patterns from a mixed pattern once it has been appropriately trained to convergence. To test for the robustness of this architecture the number of individual patterns to be trained on has been increased to four in the next experiment.

Experiment 2. With the successful recognition of two individual species from their mixed patterns, the modified mlp architecture was trained on spectra from three individual plasmas - hydrogen, argon, and methane. The individual patterns that the network is being trained on represent the species - Ar, H, CH, CH^+.

Table 2. Results from experiment 2 - species identification

files	sp1	sp2	sp3	sp4	sp5	sp11	sp12	sp13
Ar	0.37	0.24	0.24	0.25	0.28	0.28	0.29	0.26
H	0.96	0.97	0.97	0.96	0.97	0.96	0.97	0.97
files	sp14	sp18	sp19	hydr4	hydr5	hydr6	hydr7	hydr8
Ar	0.30	0.27	0.27	0.18	0.20	0.22	0.18	0.13
H	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97
files	hydr9							
Ar	0.17					(17 hydrogen plasmas)		
H	0.97							
files	arg2	arg3	arg4	arg5	arg6	arg7	(6 argon plasmas)	
Ar	0.97	0.97	0.97	0.97	0.96	0.97		
H	***0.96***	***0.96***	***0.96***	***0.97***	***0.95***	***0.96***		
files	sp6	sp7	sp8	sp9	sp10	sp15	sp16	sp17
Ar	***0.49***	***0.38***	0.63	0.67	0.69	0.58	0.71	***0.49***
H	0.97	0.97	0.97	0.97	0.97	0.97	0.97	0.97
files	sp20	sp21	arhy1	arhy2	arhy3	arhy4	arhy5	(15 mixed
Ar	0.54	***0.42***	0.97	0.96	0.97	0.96	0.96	Ar/H2
H	0.97	0.97	0.95	0.96	0.96	0.96	0.96	plasmas)
files	NI9_3	NI9_5	NI9_7	NI9_9	NI9_11	(5 mixed plasmas		
Ar	0.92	0.90	0.92	0.93	0.93	methane/hydrogen/argon)		
H	0.94	0.94	0.93	0.94	0.94			
CH	0.92	0.93	0.85	0.95	0.93			
CH+	0.96	0.89	0.89	0.91	0.91			

The spectra from a new batch of seventeen hydrogen plasmas, six argon plasmas and fifteen mixed argon/hydrogen plasmas were tested on the trained network, and once again from the results shown in Table 2, the neural network recognises both argon and hydrogen in the mixed spectra.

The argon plasmas did contain atomic hydrogen (bold italic highlights). However, with the threshold value of 0.5 the output values from four of the fifteen Ar/H_2 test set (bold italic highlights) would be interpreted as 'Atomic argon, Ar, Absent'. The main conclusion drawn from these results is -

The network is not robust enough when it comes to the recognition of individual patterns in mixed patterns due to the change in the peak pattern of certain chemical species within the emission spectrum from a mixed plasma. This is addressed more clearly in experiment 3, indicating the next steps to apply a simple rule-based system with qualitative reasoning principles, some of which have been implemented at the training of the neural network and can be incorporated into the rule-base.

Experiment 3. 104 optical emission spectral patterns from real plasmas have been selected to train a different architecture of 38:7:7 i.e. a network with 38 input neurons, 7 hidden, and 7 output neurons (each output neuron to recognise one of the seven species: Ar, H, H_2, N_2, N_2^+, CH, CH^+). All selected features are used as inputs to train the network. These features are the normalised intensities at the relevant wavelength bands that represent the unique peak pattern of the seven individual species plus the plasma parameter conditions for each spectra. After successful training for 104 epochs, the network is tested on a different set of 19 real spectra.

Table 3. Results of experiment 3 - accuracy in species identification

plasma type	Ar	H	H2	N2	N2+	CH	CH+	Accuracy
Ar only	1	0	0	0	0	0	0	100%
H2 only	1	1	1	0	0	0	0	Ar present
H2 only	1	1	1	0	0	0	0	Ar present
N2 only	0	0	0	1	1	0	0	100%
CH4 only	1	1	1	0	0	1	1	Ar present
Ar/H2	1	***0***	***0***	0	0	0	0	H2 absent
Ar/H2	1	1	1	0	0	0	0	100%
Ar/H2	***0***	1	1	0	0	0	0	Ar absent
Ar/N2	1	1	1	1	1	0	0	H2 present
H2/N2	0	1	1	1	1	0	0	100%
Ar/H2/N2	0	1	1	1	1	0	0	Ar absent
Ar/H2/N2	1	1	1	1	1	0	0	100%
Ar/H2/CH4	1	1	1	0	0	1	1	100%
Ar/H2/CH4	1	1	1	0	0	1	1	100%
Ar/H2/CH4	1	1	1	0	0	1	1	100%
Ar/H2/CH4/N2	1	1	1	1	1	1	1	100%
Ar/H2/CH4/N2	1	1	1	1	1	1	1	100%
Ar/H2/CH4/N2	1	1	1	1	1	1	1	100%
Ar/H2/CH4/N2	1	1	1	1	1	1	1	100%

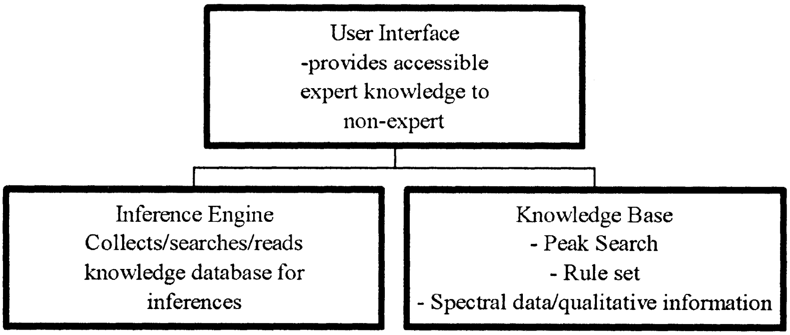
The results from the test set of 19 spectra shown in Table 3 confirm three points. Note that the output values are rounded up to the nearest decimal i.e. 0 or 1 for viewing ease.

1. A 3-layer multilayer perceptron neural network trained using the back-propagation algorithm can identify the presence of individual species within a mixed pattern of species (spectra from mixed gas plasmas) when trained on a sufficient set of both single and mixed patterns.
2. It can detect other species that were existent within the plasma at the time of OES data collection but not necessarily known to be present in that specific system, therefore the interpretation is that the network can detect impurities or contaminant species.
3. The network output results from two of the mixed pattern set from spectra of argon/hydrogen plasmas (bold italic highlights), show that occasionally the network may fail in species detection. This has raised the issue of other external parameters affecting the network's performance and so it is important to incorporate some expert knowledge of optical emission spectroscopy to correctly interpret the OES data. This is addressed in the rule-based system.

2.2 Rule-based System

A prototype rule-based system has been developed to explicitly characterise seven different chemical species by using sections of the unique peak pattern feature extraction process (implemented in the neural network architecture) to identify a database of spectral peak searches. The database consists of facts and rules expressed as knowledge representations that are declared into the knowledge base, the inference engine searches the knowledge base for species presence or absence and can respond to the user via the user interface (see Fig. 2 for the component parts of the rule-based expert system). Figures 3 and 4 show portions of the knowledge base peak search, and part of ruleset containing thresholds to establish presence or absence of a species.

Fig. 2. Component parts of the rule-based expert system



The knowledge base accepts spectral data in a normalised format and the rule base is hand-crafted to pick out any primary peaks (e.g. H_Pline), secondary peaks (e.g. H_Sline) and/or tertiary peaks (e.g. H_Cline) by setting arbitrary threshold values, in order to confirm the presence or absence of species. A randomly selected test set of ten spectral outputs has been analysed using this rule-based system with successful responses to the user i.e. ten correct responses to each of the ten spectral data sets explicitly indicating which chemical species are present. The current knowledge base can be readily modified to include further expert knowledge to incorporate a wider and quicker spectral search and identification of more chemical species for automating the process. In the process of achieving this, the implementation of qualitative reasoning to interpret inconclusive or ambiguous responses from the neural network and address an incomplete ruleset within the rule-based system is being applied.

Fig. 3. Portion of knowledge base - peak search

```
                /* peak_search*/

peak_search('Input the normalised
       intensity value at the 750-nm
       bandhead:', read(Ar_Pline)).

peak_search('Input the normalised
       intensity value at the 656-nm
       bandhead:', read(Ar_Pline)).
```

Fig. 4. Part of Knowledge base - Ruleset which contains thresholds to establish presence or absence of a species

```
/* Rules, incorporating threshold for normalised
   intensity values for a given emission line */

rule(1, atomic_argon_present):-
  emission_line(Ar_Pline > 0.6, Ar_Sline > 0.3,
     Ar_Cline > 0.1);
  emission_line(Ar_Pline > 0.6, Ar_Sline > 0.3);
  emission_line(Ar_Pline > 0.6) .

rule(2, atomic_hydrogen_present):-
  emission_line(H_Pline > 0.7, H_Sline > 0.4,
     Ar_Cline > 0.1);
  emission_line(H_Pline > 0.7, H_Sline > 0.4);
  emission_line(H_Pline > 0.7) .
```

2.3 Final Hybrid Approach

However well-trodden the work-path of combining rule-based systems, neural network approaches and several other techniques for data classification, prediction or retrieval/detection may be, this work deviates from what has gone before in so far as its compelling end-result uses successfully trained ANN models of individual chemical species (from optical emission spectra) to generate a prolific set of semantic rules (portion of a set list in Table 4). The ANN models for each species was created by utilising the plasma process parameters which consist of gas flowrates, pressure and power as the inputs to the network and the target outputs were the actual emission intensities at the unique peak positions for the particular species being modeled. Fig. 5 shows the neural network architecture of the species model (for atomic argon) which consists of a 3-layer fully connected feedforward network using a *linear output* function to cater for the large intensity range, and the typical sigmoidal function on the hidden layer. A selection of several varying plasma spectra were used in the training of the network models. The seven species models have been tested on different sets of spectra for verification with very good performance results (Table 5 shows excellent performance on testing the argon species model). Work continues to produce a generic hybrid system which adopts the premise that a trained neural network model that accepts continuous valued inputs can produce explicit relationships between input patterns and trained responses.

Table 4. Portion of condition set list from trained ANN models

Condition 1. *If (Pressure and Argon flowrate are HIGH) then atomic argon (Ar) is detected.*
[Ar420 is LOW, Ar750 is MEDIUM, Ar763 is HIGH.]

Condition 2. *If (Power and Hydrogen flowrate are HIGH) then atomic hydrogen (H) is detected.*
[All three H-lines are present, H434 is MEDIUM, H486 is LOW, H656 is HIGH.]

...

Condition 4. *If (Nitrogen flowrate is HIGH) then molecular nitrogen (N_2) is detected.*
[N_2(337) is HIGH, N_2(389) is LOW].

Table 5. Results from a test set of the ANN model for atomic argon, Ar

	target output			network output		
plasma	Ar(420)	Ar(750)	Ar(763)	Ar(420)	Ar(750)	Ar(763)
H2	654	41	23	900	146	90
Ar/H2	510	227	100	512	217	121
	679	212	123	693	224	130
Ar	214	433	240	241	653	387
Ar/H2/N2	178	270	184	87	236	129
Ar/N2	179	356	194	91	313	182
Ar/H/CH4	182	519	326	163	471	265
Ar/H/N/CH	272	299	131	199	359	198

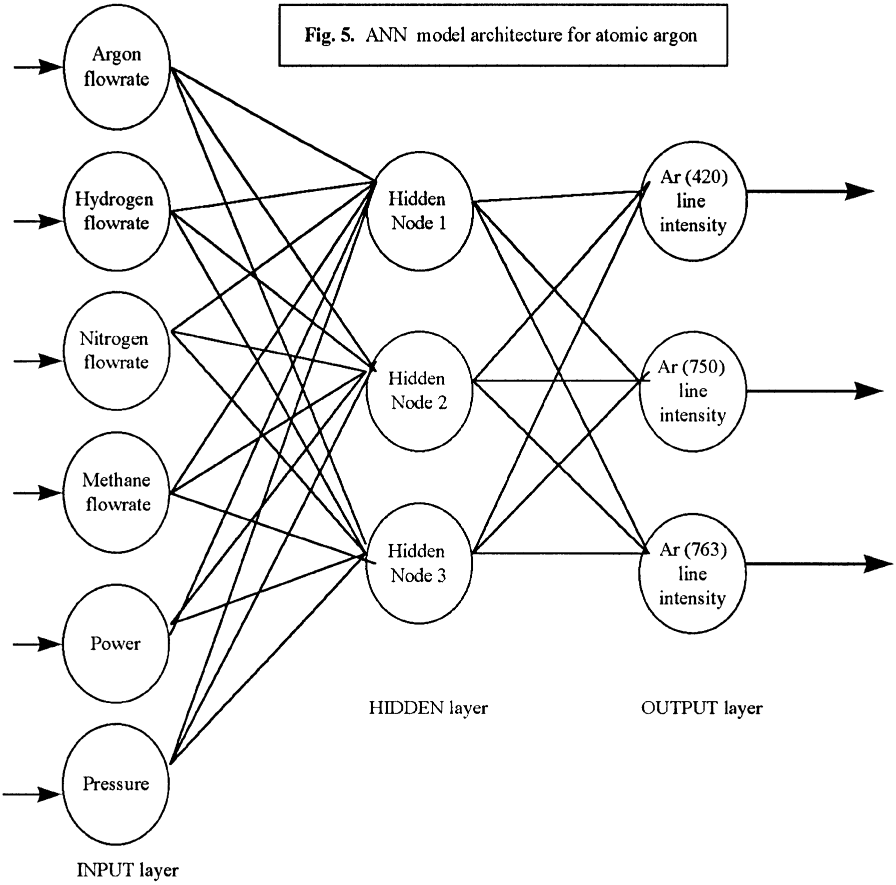


Fig. 5. ANN model architecture for atomic argon

2.3 Future Work

Various multivariate techniques including principal component analysis (PCA) and partial least squares (PLS) [11] are currently employed to analyse spectral data which are typically renowned for the inherent difficulties in their prediction due to the large number of variables involved. Therefore, the goal of this work was not to reiterate the application of such multivariate techniques to optical emission spectral analysis but to apply artificial intelligence techniques instead. The work to follow from the results obtained will incorporate qualitative reasoning [12], [13] principles to address some of the external parameters that influence complex optical emission spectral analysis.

Conclusions

The AI techniques employed have been successfully incorporated into a hybrid system of neural networks and rule-based system that has produced a semi-automated chemical species detection unit that provides simple and fast interpretation of ex-situ optical emission spectra stored in a digitised format. Future work will adapt the hybrid system by extraction of its generic properties to make it transferable to a much wider knowledge base of spectral data.

References

1. Malchow, D.S.: Characterization of plasma processes with optical emission spectroscopy. SPIE. **1392** (1990) 498-505
2. Clay, K.J., Speakman S.P., Amaratunga, G.A.J and Silva S.R.P.: Characterization of a-C:H:N deposition from CH_4/N_2 rf plasmas using optical emission spectroscopy. J.Appl.Phys. **79** (1996) 7227-7233
3. Bockel S., Amorim J., Baravian G., Ricard, A. and Stratil P.: A spectroscopic study of active species in DC and HF flowing discharges in N_2-H_2 and Ar-N_2-H_2 mixtures. Plasma Sources Sci. Technol. **5** (1996) 567-572
4. Gifford, G.G.: Applications of optical emission spectroscopy in plasma manufacturing systems. SPIE. **1392** (1990) 454-465
5. Affolter, C., Baumann, K., Clerc, J.-T., Schriber, H., and Pretsch, E.: Automatic interpretation of infrared spectra. Mikrochim. Acta [Suppl.]. **14** (1997) 143-147.
6. Mittermayr, C.R., Drouen, A.C.J.H., Otto, M., and Grasserbauer, M.: Neural networks for library search of ultraviolet spectra. Analytica Chimica Acta. **294** (1994) 227-242
7. Picton, P.D., Hopgood, A.A., Braithwaite, N.S., Phillips H.J.: Identifying chemical species in a plasma using a neural network. Proc. IEA/AIE, Fukuoka, Japan (1996) 593-598
8. Hopgood, A.A., Phillips, H.J., Picton, P.D., and Braithwaite N.S.: Fuzzy Logic in a Blackboard System for controlling Plasma Deposition Processes. Accepted for publication in AI in Engineering (1997)
9. Bell, S.E., and Mead, W.C.: Artificial Intelligence and neural networks applied to ion mobility spectrometry. Computer-Enhanced Analytical Spectroscopy. Vol.4 (1993) 275-303
10. Henson, T., Huxhold, W. and Bowman, D.: An enhanced neural network learning algorithm with simulated annealing. SPIE 3rd Workshop on NNs **1721** (1992) 87-94
11. Niemczyk, T.M., Franke, J.E., Wangmaneerat, B., Chen, C.S., and Haaland, D.M.: Extracting Qualitative and Quantitative information from infrared emission spectral data. Computer-Enhanced Analytical Spectroscopy. Vol.4 (1993) 165-185
12. Zhao, Qi, and Nishida, T.: Qualitative Decomposition and Recognition of Infrared spectra. IEICE Trans. Inf. & Sys. **E79-D** (1996) 881-887
13. Nishida, T. and Zhao, Qi.: Using Qualitative Hypotheses to Identify Inaccurate Data. JAIR (1995) 119-145

KADS Qualitative Model Based Motor Speed Control

J. Ruiz Gomez

Departamento de Ingenieria de Sistemas y
Automatica
Universidad de Malaga
ruizg@ ctima.uma.es

Abstract. The aim of this work is to propose a qualitative modelling method based on two AI approaches: Qualitative Physics and KADS. It is an expertise data driven modelling, that arising from the domain ontology and dynamics achieves a symbolic formalization that models a dynamical system. The system model is used for a control application (also based on KADS): a D.C. motor speed control.

1 Introduction

Some systems commonly considered as "mathematizable" ones (i.e. mechanical or electrical systems) frequently exceed in complexity, and their management is conditioned by the resolution of differential equations, resolution usually difficult when systems are no-linear ones. On the other hand, it is in poor structured systems where KBS are usually applied. So it looks like that in these two cases a qualitative methodology could bring some simplifying resolution ways.

Knowledge engineering has been gradually developing from a human expert extraction process towards a new approach [5] that copes with knowledge acquisition as a modelling activity. In this situation, a KBS development methodologies as KADS (Knowledge Acquisition and Decision Support) can contribute to the methodological background of this modelling activity.

Merging two well known AI approaches: Qualitative Physics settled by Bobrow [3], Forbus [13], De Kleer [10] and Kuipers [20], and the KADS methodology of Wielinga et al. [5] a flexible qualitative methodology to obtain domain models of dynamic systems is developed in this work. It is a hybrid methodology able to cope with poorly structured dynamic systems. Input and output signals are used to identify the system. It is presumed that the amount of data is enough to cope with the modelling process. These data have been taken in a steady equilibrium state.

The paper layout is: first an introduction and an overview of related works, afterwards the proposed approach description and finally an engineering application to the modelling and control of a DC motor velocity.

2 Related Works

The first foundations of the proposed modelling method are KADS and CommomKADS [16, 26] developed by Wiellinga and co-workers [3, 6, 30] as an integrated approach for the development of Knowledge Based Systems. The second one is Qualitative Reasoning, developed by Bobrow [3] De Kleer [10, 31] and Kuipers [20, 21]. A global overview of the qualitative approach, besides System modelling and its applications to several areas, can be seen in [9] and in a reduced way in [4]. The qualitative reasoning mathematical foundations have been treated by Iwasaky, Simon and Kalagnanam [18, 19]. Zeigler [32] in a classical work settled modelling foundations as an area-independent approach.

Arising from KADS, the domain knowledge modelling has been studied by Abu-Hanna and Hansweijer [1]. Voss and Karbach [29] cope with expertise models implementation, and Neale [23] have centred his work in obtaining a conceptual model that works as a bridge between verbal data and computational implementation. From the qualitative reasoning viewpoint Dohnald [11] applies qualitative mathematics to anaerobic fermentation processes.

An approach different from KADS and Qualitative Reasoning, is that of Chandrasekaran [7, 8] who tries to model knowledge by loosing the modelling process of its link with determined goal or tasks, and considering this modelling process as a task-structure. In automation areas, Leitch and co-workers have proposed a task classification by means of generic tasks for KBS development [22].

Finally, the conceptualisation of specifications, in the knowledge implantation accomplished by means of computational models, has been developed from the viewpoint of ontology by Gruber [15] and Guarino [14].

3 Towards a Flexible Qualitative Modelling Methodology

This paper aim is to obtain a domain model of a dynamical system. It's intended to give a methodology [15,16] to obtain the domain model knowledge, composed by a dynamical core elicited as rules.

At first KADS divided the expertise [17] in a four layers model: domain, inference, task and strategy layers. The inner one, the domain knowledge layer, is considered in a different way from KADS: here the domain Ontology is only a part of the Domain Model.

The qualitative modelling of domain knowledge for a continuous/discrete time dynamic system can be made in two steps: a) to determine the static features or domain ontology, and b) to obtain the rules that describe the dynamical side. The modelling process can be summarised in the following Figure 1:

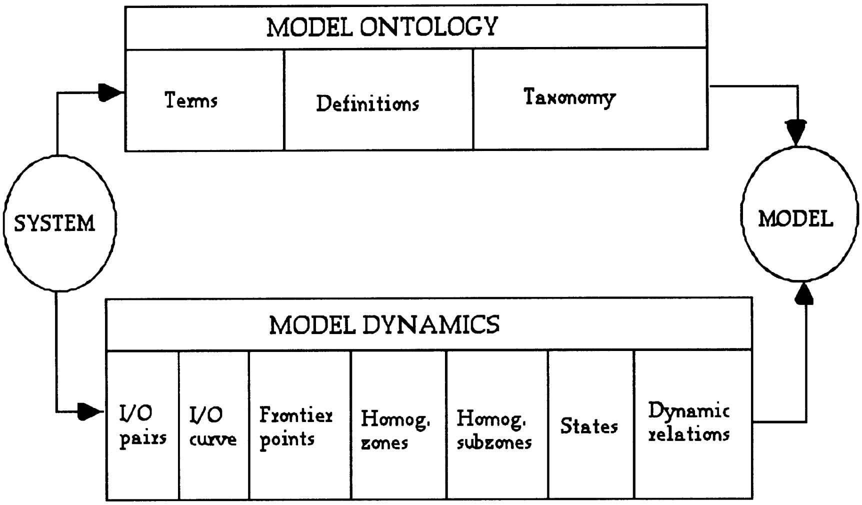


Fig. 1 Qualitative modelling process.

3.1 Domain Ontology

The Domain Ontology is defined by means of three sets: a set of terms, a set that contains their definitions (typology) and a third set of relations between the terms (taxonomy).

When a dynamical system is observed some objects or elements can be identified. These objects are the *terms,* and constitute the system lexicon. Some of these terms are: *plant, variable, input-variable, value, cloud_of_points, state, segment, input-segment, sample-period, input-range.*

After listing all the terms that establish the domain it is necessary to explain their definitions (*typology).* These definitions are not explained here in a strict formal sense, but in a semiformal and intuitive way. Examples of these definitions are:

- *variable*: a magnitude (pressure, temperature,...) that can take values on real numbers x∈R.
- *range:* variation interval of a variable, set of all the possible values of the variable, X={x1, x2, x3,...,xn}.
- *point:* a pair of constant values (x, y) where x belongs to input set X and y to output set Y.
- *cloud_of_points*: a finite set of points {(input-value, output-value)} experimentally obtained.
- *critical-point:* a point of zero or non existing derivative.
- *segment*: interval containing a finite set of a variable values e X= {xl, x2, x3, ...,xn}

Some of these terms are common in mathematics, but others are defined for this methodology, as *input/output-curve, homogenous-zone, subhomogenous-zone, frontier-point* and *state.*

The set of points or *cloud_of_points* "is seen" as continuous curve, a curve generated by means of a pattern recognition process achieved by the user experience from the I/O points graph. This recognition of a set of points as a continuous curve shows the importance of the subjective aspect of the process. When modelling the domain knowledge the user prephysical knowledge common sense becomes apparent.

A variable determines through its variation pattern the existence of a state. The states are determined by the observation of the qualitative influence of input magnitudes over output ones.

To determine the states, three previous concepts are needed: *frontier-point, homogenous-zone and homogenous-subzone.*

As *frontier-points* we take the abscissa of points that separate two adjacent zones in which certain uniformity or homogeneity is observed. In a mathematical sense these points will be that of zero function value, or a critical points (points without derivative or with zero value derivative). These points can be of three types: discontinuity, maximum or minimum, or angular points.

We consider as *homogenous-zone* a region of the input/output curve with monotone variation (not increasing or not decreasing curve region). When a *homogenous-zone* is split in new monotone regions, *homogenous-subzones* are obtained.

By means of these two concepts of *frontier-point* and *homogenous-zone* it is possible to divide the input/output curve in several regions with a single variation mode so as to allow the same treatment inside these regions.

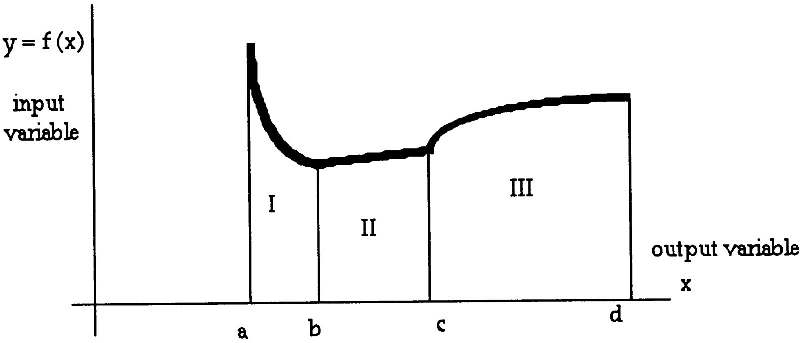


Fig 2 Input/output curve *homogenous_zones.*

State changes occur at *frontier-points.* Every *homogenous-zone* or *homogenous-subzones* defines a *state.* Let us suppose the input/output curve of Figure 2. Three different regions can be found, which generate three *homogenous-zones (I, II and III).*

Last, the Domain Taxonomy collects a terms classification of the type " is-a" hierarchical relations, as: frontier *is-part* of input-range.

3.2 Domain Dynamics

After getting the Domain Ontology, the domain dynamical component must be obtained. This dynamic aspect copes with the system input/output behaviour, that is stated as a set of production rules.

The input/output curve represents a global relation or a group of partial I/O relations. The Domain Knowledge dynamics is gathered from these relations between input ranges and output ranges.

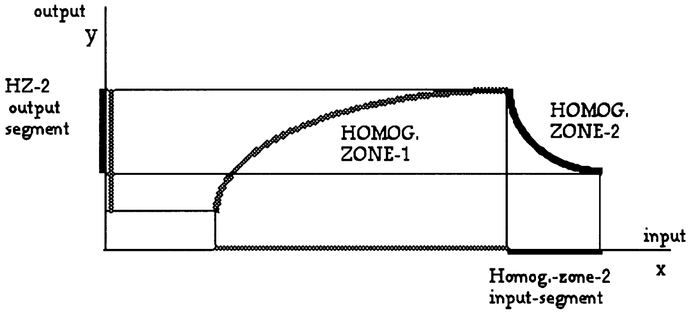


Fig. 3 Correspondence relations for *homogenous_zones.*

Let us suppose a homogenous_zone partition as that of Figure 3. Relations between corresponding input and output segments could be assessed as:

- R1: *Input_value* contained in *input_segment_HZ1* implies that *output_value* is contained in *output_segment* of HZ1.
- R2: *Input_value* contained in *input_segment_HZ2* implies that *output_value* is contained in *output_segment* of HZ2.

Following a splitting iterative process for these homogenous zones allows us to increase the accuracy of the relations.

3.3 Domain Model

When both Domain Ontology and Dynamics have been obtained, a set of formal settlements are achieved. These settlements establish the *domain model* nucleus.

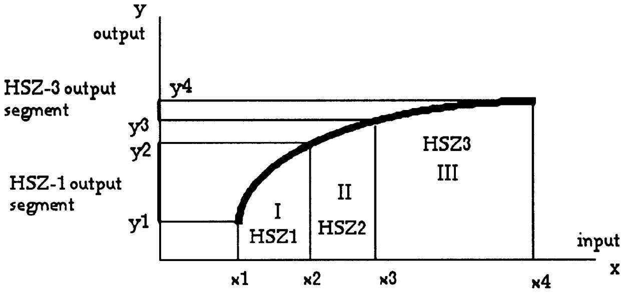


Fig. 4 Splitting of an *homogenous_zone* in three *subhomogenous_zones.*

Input/output relations depicted in Figure 4 can be written as the following rules:

A) For all the homogenous-subzones:

- CAUSAL RELATION: null input.

No input => the output remains without any change.

RULE: IF *input_value* = 0 THEN *output_value* doesn't change.

- BEHAVIOURAL RELATION: input out of range.

If x varies outside its variation range ("out of the curve") => y is not affected (remains constant)

RULES:

IF x increases & ($x < x1$ or $x > x4$) *THEN* the *output* is constant and equal to the previous value.

IF x decreases & ($x < x1$ or $x > x4$) THEN the *output* is constant and equal to the previous value.

- BEHAVIOURAL RELATION:

Monotonicity: if x increases e => y increases.

RULE: IF *input_value* increases THEN *output_value* increases

B) For *homogenous_subzone_1*:

- BEHAVIOURAL RELATION:

Input_value contained in *input_segment_SZH1* implies *output_value* is contained in *output_segment* of SHZ1.

RULE: IF $x1 < x < x2$ THEN $y1 < y < y2$

c) And in a similar way on all the remaining *homogenous_subzones.*

It can be observed that from a certain homogenous_zone several rules can be obtained.

4 Motor Modelling

The modelling proposed method has been put to work in the speed control of a DC motor. The experimental set is constituted by a preamplifier, a D.C. motor, an analogical tachometer, a photoelectic speed sensor, an input/output card and a PC-486 computer.

In this application two different tasks are accomplished:

1) First the qualitative modelling process of the physical device (preamplifier and motor) or domain model.

2) Second, the control implementation arising from the obtained domain model. This implementation also used KADS methodology.

4.1 Domain Model

First the domain ontology must be obtained.

Some of the more significant terms, including their definition, are:

plant:	*the motor system (preamplifier, DC motor)*	
input_variable:	*electric voltage*	*real number*
output_variable:	*motor velocity,* r.p.m	*real number*
input_range:	*2.8 to 5 volts*	*real number*
output_range:	*600 to 3400 r.p.m.*	*real number*
sample_period:	2 *sec.*	*real number*
cloud_of_points:	*pairs(x, y)*	*(real number, real number)*

The experimentally measured data are shown in Figure 5.

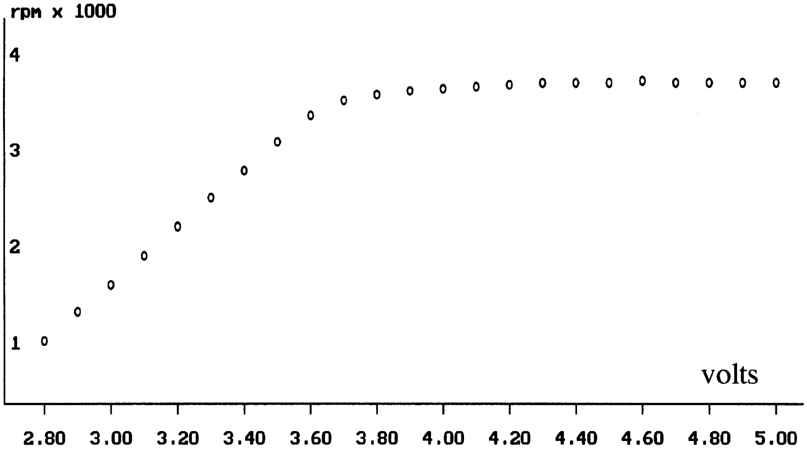


Fig. 5 Input/output motor experimental *cloud of points.*

From this cloud_of_points we can perform several splitting levels. At first level only four homogenous_zones are obtained:

- HZ1: for input values between 5.0 and 4.9 the output values are in the segment 3343-3386.
- HZ2: for input values between 4.9 and 4.4 the output values are in the segment 3386-3310.
- HZ3: for input values between 4.4 and 4.3 the output values are in the segment 310-3316.
- HZ4: for input values between 4.4 and 2.8 the output values are in the segment 3316-643.
-

The *domain dynamics* is given by the corresponding four rules:

- R1.- IF 5 ≤ Vi ≤ 4.9 THEN 3343 ≤ Vo ≤ 3386
- R2.- IF 4.9 ≤ Vi ≤ 4.8 THEN 3386≤ Vo ≤3310
- R3.- IF 4.8 ≤ Vi ≤ 4.7 THEN 3310 ≤ Vo ≤3316
- R4.- IF 4.7 ≤ Vi ≤ 4.6 THEN 3316≤ Vo ≤ 643

At the third splitting level the rules generated are ten, and when working at the fifth level fourteen rules are got.

4.2 Controlled System Response

To validate models a control application has been chosen. To implement this control application KADS methodology has been used again. Following KADS, the Application Knowledge must be obtained. So we must get Inference, Task and Strategy Domain layers.

To get the system response step signals have been used. Figure 6 shows the output signal for a desired output of 2500 r.p.m. by using the third level of splitting resulting in ten rules.

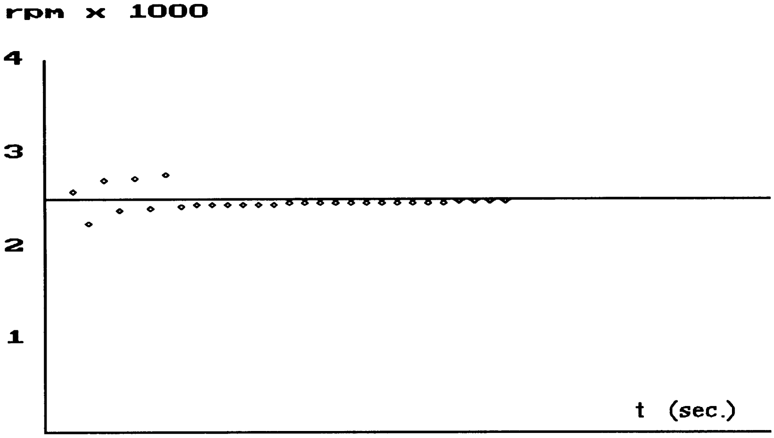


Fig. 6 Response of the model of 10 rules and level 3 to a step of 2500 r.p.m.

Figure 7 shows the output to the same input, but with the fifth partition level (resulting in 14 rules):

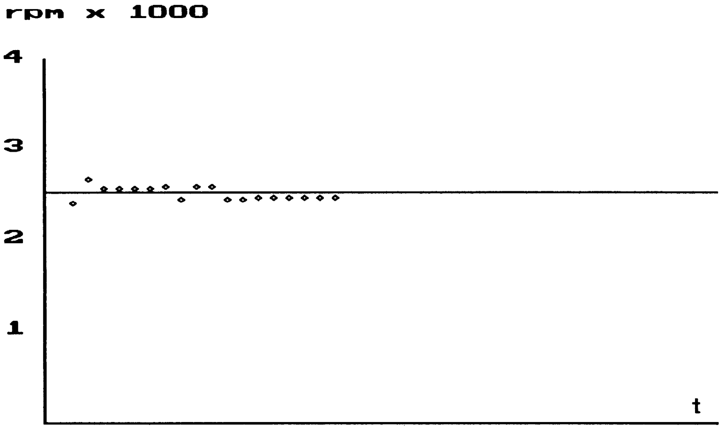


Fig. 7 Response of the model of14 rules and level 5 to the same step of 2500 r.p.m.

5 Concluding Remarks

The proposed modelling methodology has allowed to obtain a symbolic non quantitative model of the domain arising from observation and experimental measurements of a system. We have coped with a dynamical system without considering its linearity or non-linearity.

In every modelling process several models are obtained, so that the user can chose between them. This feature allows non-expert on dynamic systems theory to perform modelling and control.

The modelling method have been implemented in an application: real time speed control for an electric D.C. motor. Models obtained have been used in a control application and have allowed us to achieve acceptable results, close to those obtained by means of conventional mathematical modelling and control methods.

A shortcoming in the control application is a trend to instability found in several cases. A way to cope with these stability problems will possibly be by using fuzzy sets instead of crisp ones in the dynamic part of the models.

References

1. Abu-Hanna, A.; Jansweijer, W. (1994): Modeling Domain Knowledge Using Explicit Conceptualization. *IEEE Expert*, Oct. 53-64
2. Bigot, C. (1994): *CommomKADS Glossary*. ESPRIT Project P5248 KADS-II Document.The KADS-II Consortium. Amsterdam.
3. Bobrow, D.G(1984): *Qualitative Reasoning about Physical Systems*. Amsterdam. North-Holland.
4. Bonissone, P.; Valavanis, K.P.(1985): A Compartive Study of different Approaches to Qualitative Physics Theories, en Weisbin (ed): *Artif. Intelligence Applications. The Engineering of Knowledge-Based Systems*. Washington. IEEE Soc. Press.
5. Breuker, J.; Wielinga, B.(1989): Models of Expertise in Knowledge Adquisition, Guida, G.; Tasso, C.(ed.):*Topics in Expert Systems, Design, Methodologies and Tools*. Amsterdam. North-Holland.
6. Breuker, J. ; Van De Velde, W. (ed.) (1994): *Expertise Model Document Part II: The CommonKADS Library*. ESPRIT Project P5248 KADS-II Document. Univ. Brussels.
7. Chandrasekaran, B. (1986): Generic tasks in knowledge-based reasoning: High level building blocks for expert systems design *IEEE Expert* 1 (3). 23-30
8. Chandrasekaran, B. et al. (1992): Task-Structure Analysis for Knowledge Modeling, *Communictions ACM* 35 (9). 124-137.
9. Dague, P y MQ&D. (1995): Qualitative Reasoning: A Survey of Techniques and Applications, *AI Communications* 8. 117-206.
10. De Kleer, J.;Brown, J.S.(1983):A Qualitative Physics Baseds on Confluences, en Bobrow D.G. (ed): Qualitative Reasoning about Physical Systems. Amsterdam. North-Holland.
11. Dohnal, M. (1991): A methodology for common-sense model development. *Computers in Industry* 16. 141-158.
12. Fernandez González, R.; Ruiz Gomez, J. (1996): Modelización Cualitativa de Sistemas Dinámicos. http://147.96.41.128/ (in spanish)
13. Forbus, K.(1993): Qualitative process theory: twelve years after. *Artif. Intellig.* 59. 115-123.
14. Guarino N., Giaretta P.(1994): Ontologies and Knowledge Bases: Towards a Terminological Clarification, Doyle, N. J., Sandewall, E. and Torasso, P. (eds.): *Principles of Knowledge Representation and Reasoning*. Proc. 4th Int. Conf. KR94. Morgan Kaufmann. San Mateo. CA.

15. Gruber, T.; Olsen, G.(1994): An Ontology for Engineering Mathematics, Doyle, N.J. Sandewall, E and Torasso, P. (eds): Principles of Knowledge Representation and Reasoning. Reasoning. Proc. 4th Int. Conf. KR94. Morgan Kaufmann.San Mateo.CA
16. Gustafsson, M.; Menezes, W. (1994): *CommonKADS Reference Manual.* ESPRIT Project P5248 KADS-II Document. Univ. Amsterdam.
17. Hickman, F. et al. (1989): *Analysis For Knowledge-Based Systems.* New York. Ellis Horwood.
18. Iwasaki, Y.; Simon, H. (1993): Retrospective On 'Causality In Device Behavior'. *Artif. Intelligence* 59. 141-146.
19. Kalagnanam, J. ; Simon, H. ; Iwasaki, Y. (1991): The Matematical Bases For Qualitative Reasoning. *IEEE Expert*, april. 11-19.
20. Kuipers, B. (1984): Commonsense Reasoning About Causality: Deriving Behavior From Structure.*Artif. Intelligence* 24. 169-203.
21. Kuipers, B. (1993): Reasoning With Qualitative Models. *Artif. Intelligence* 59. 125-132.
22. Leitch, R.; Gallanti, M. (1992): Task Classification for Knowledge-Based Systems in Industrial Automation. *IEEE Transactions on Systems, Man, and Cybernetics,* 22 (1) 142-152.
23. Neale,I.M.(1990): Modelling Expertise for KBS Development. *J. Opl. Res. Soc.* 41 (5). 447-458.
24. Ruiz Gomez, J. and Fernandez Gonzalez, R. (1997): "Modelizaci6n cualitativa basada en KADS: Una aplicaci6n a control de temperatura. Proceed. CAEPIA'97. Malaga. Nov. 65-74.
25. Ruiz Gomez, J. (1996): *Modelización Cualitativa de Sistemas Dinámicos.* Ph. D. Thesis. Univ. Complutense. Madrid. (in spanish)
26. Schreiber, G. et al.(1994): CommonKADS: A Comprehensive Methodology for KBS Development. *IEEE Expert, Dec.* 28-36.
27. Shen, Q.; Leitch, R.; Steele, A.D. (1994): A Generic Harness for the Systematic Generation of Multiple Models, *Proceedings of the Eigth International Qualitative Reasoning about Physical Systems Workshop.* Nara. Japan.
28. Top, J. ; Akkermans, H. (1993): *Engineering Modelling- A Contribution to the CommonKADS Library*. ESPRIT Project P5248 KADS-II Document.
29. Voss, A. ; Karbach, W.(1993): Implementing KADS Expertise Models with Model-K. *IEEE Expert*, Aug. 74-81.
30. Wielinga, B. J. (ed.) (1994): *Expertise Model Definition Document.* ESPRIT Project P5248 KADS-II Document. Univ. Amsterdam.
31. Williams, B. ; De Kleer, J. (1991): Qualitative Reasoning About Physical Systems: A Return To Roots. *Artif. Intelligence* 51. 1-9.
32. Zeigler, B. P. (1976): *Theory Of Modelling And Simulation.* London. John Wiley & Sons.

Negligibility Relations between Real Numbers and Qualitative Labels

Mónica Sánchez, Francesc Prats, and Núria Piera

Dept. Matemàtica Aplicada II
FIB-Universitat Politècnica de Catalunya
08028 Barcelona, Spain
LEA-SICA
(European Associated Laboratory of
Intelligent Systems and Advanced Control)
{Monicas, Prats}@ma2.upc.es

Abstract. This work continues previous ones in the study of a relative orders of magnitude model, based in the formalization of certain qualitative knowledge between real quantities and qualitative labels. The mathematical model for negligibility in qualitative reasoning presented is to be used in the not unusual situation in which one disposes of some real quantitative data and only disposes of the qualitative descriptions of other ones. Negligibility relations between real numbers, between numbers and labels and between labels are defined and studied with detail.

1 Introduction

The frequent lack of quantitative information, and sometimes also the "excess" of quantitative information, together with the confirmation that it is possible, in certain situations, to arrive at significative conclusions from a purely qualitative knowledge, makes clear the importance of the research of qualitative representations and qualitative techniques in several scientific fields.

This paper continues and concludes previous works in the study of a relative orders of magnitude model, based in the formalization of certain qualitative knowledge between real quantities often used in Engineering. The qualitative knowledge considered is relational knowledge of the kind: two quantities have the same order of magnitude, or are close to each other, or one of them is negligible with respect to the other.

The first formalizations [7], [4], [1,2,3] for relative orders of magnitude models were based on the sign algebra, where quantities are given by their signs. In [San 96a] this qualitative relational knowledge is formalized in another algebra, where the labels that describe the quantities are more precise than the simple signs, i.e., in the Qualitative Order of Magnitude Algebra. Over this algebra we obtained a new relative qualitative model: the relations of having the same order magnitude, negligibility and proximity between real quantities were formalized in such a way that the given axiomatization is compatible with the particular label language of the Absolute Orders of Magnitude Models (see [11], [6]).

The idea now is to solve the not unusual situation in which one disposes of some real quantitative data and only disposes of the qualitative descriptions of other ones. If it is necessary to work simultaneously with both kinds of data, it is necessary first to define and study certain binary relations on the one hand between real numbers and Q-labels, on the other between Q-labels and Q-labels, that correspond to the ideas of negligibility, same order of magnitude and proximity, studied for real numbers. This will allow the construction of an automatic system where it can be possible to reason in both qualitative and quantitative ways.

2 Frame of the work

The Quantity Space used here, i.e., the set made up of the qualitative labels that can be used in order to describe the objects, is constructed from the partition of the real line given by two real numbers α, β, with $0 < \alpha < \beta$:

$(-\infty, -\beta], (-\beta, -\alpha], (-\alpha, 0), \{0\}, (0, \alpha), [\alpha, \beta), [\beta, +\infty)$

It is denoted by S_1 the set formed by the labels:

$$S_1 = \{NL, NM, NS, [0], PS, PM, PL\}.$$

Each one of these labels is associated to the corresponding interval of the real line.

On the other hand, given two elements E_1, E_2 in $S_1 - \{0\}$, *the label* $[E_1, E_2]$ is defined as the smallest interval of the real line, with respect to the inclusion, that contains E_1, E_2. For example, given $NM, PS \in S_1$, the label $[NM, PS]$ is the real interval whose extrems are $-\alpha$ and β.

The *Complete Universe of Description based on* S_1, that is the Quantity Space of this work, is the set S:

$$S = S_1 \cup \{[E_1, E_2];\ E_1, E_2 \in S_1 - \{0\} \quad \text{and} \quad E_1 < E_2\}$$

It will be considered in S the order relation given by the set inclusion, that is,

$$\text{For}\, E_1, E_2 \in S, \quad E_1 \leq E_2 \text{ if } E_1 \subset E_2,$$

as described in [11].

It is easily seen that this order relation in S is the precision's dual relation: given $A, B \in S$, it is said that $A \leq B$ if, and only if, A and B are two qualitative descriptions of the same real number and A is more precise than B.

Finally, it is important to point out that given a real number $r \in R$, $[r]$ will denote the label of r in S_1, i.e., the element of the set S_1 that contains r, that is, the most precise qualitative description of r in this model .

The definition of negligibility given in [8] is:

Definition. Given two real numbers x and y, then x is *negligible with respect to* y, $x \,\mathrm{Ne}\, y$, if $x = y = 0$ or $y \neq 0$ and there exists some real number k such that:

$$[k] = PL \quad \text{and} \quad \left[\frac{kx}{y}\right] \leq [NS, PS].$$

It was proved that the large number k which appears in the definition of $x \,\mathrm{Ne}\, y$, $y \neq 0$, can be removed in the following way:

$$x \,\mathrm{Ne}\, y \quad \text{if and only if} \quad \left|\frac{x}{y}\right| < \frac{\alpha}{\beta}.$$

Starting from the previous definition, the relations that will be introduced below appear in the following table:

	$y \in R$	$E_2 \in S$
$x \in R$	$x\,\mathrm{Ne}\,y$	$x\,\mathrm{Ne}\,E_2$ $x\ \mathrm{Q}-\mathrm{Ne}\,E_2$
$E_1 \in S$	$E_1\,\mathrm{Ne}\,y$ $E_1\ \mathrm{Q}-\mathrm{Ne}\,y$	$E_1\,\mathrm{Ne}\,E_2$ $E_1\ \mathrm{Q}-\mathrm{Ne}\,E_2$

3 The Ne relation between numbers and labels

The relation $x\,\mathrm{Ne}\,E$ expresses the negligibility between the real number x and any number whose qualitative description is the qualitative label E:
Definition 3.1. Let x, E be a real number and a Q-label, respectively, then x *is negligible with respect to* E, $x\,\mathrm{Ne}\,E$, if

$$x\,\mathrm{Ne}\,y \quad \forall y \in E.$$

That is to say, $x\,\mathrm{Ne}\,E$ iff $E = [0]$ and $x = 0$, or, $E \neq [0]$ and $\left|\frac{x}{y}\right| < \frac{\alpha}{\beta} \quad \forall y \in R - \{0\}$ such that $y \in E$.

The possible cases for this relation are studied in the following:

Proposition 3.1.

1) If $[0] \leq E$ or NS$\leq E$ or PS $\leq E$, then $x\,\mathrm{Ne}\,E \iff x = 0$
2) If $[0]$, NS, PS $\not\leq E$, and $E \geq$ NM or PM, then $x\,\mathrm{Ne}\,E \iff |x| \leq \frac{\alpha^2}{\beta}$
3) If $E =$ NL or $E =$ PL, then $x\,\mathrm{Ne}\,E \iff |x| \leq \alpha$

Proof:

1) If $[0] \leq E$, then: $x\,\mathrm{Ne}\,E \implies x\,\mathrm{Ne}\,0 \iff x = 0$
   If NS $\leq E$ or PS $\leq E$, then it is trivial that $x\,\mathrm{Ne}\,E \iff x\,\mathrm{Ne}\,E$ PS; therefore: $x\,\mathrm{Ne}\,E \iff \left|\frac{x}{y}\right| < \frac{\alpha}{\beta} \quad \forall y \in (0,\alpha) \iff |x| < \frac{\alpha y}{\beta} \quad \forall y \in (0,\alpha) \iff x = 0$
2) If $[0]$, NS, PS $\not\leq E$, and $E \geq$ NM or PM, then it is trivial that $x\,\mathrm{Ne}\,E \iff x\,\mathrm{Ne}$ PM; therefore: $x\,\mathrm{Ne}\,E \iff \left|\frac{x}{y}\right| < \frac{\alpha}{\beta} \quad \forall y \in (\alpha,\beta) \iff |x| < \frac{\alpha y}{\beta} \quad \forall y \in (\alpha,\beta) \iff |x| \leq \frac{\alpha^2}{\beta}$
   The last equivalence is proved in the following way:
   $\Longleftarrow$) trivial, because $\alpha < y$
   $\Longrightarrow$) if we take $y_n = \alpha + \frac{1}{n}$, then $|x| < \frac{\alpha}{\beta}\left(\alpha + \frac{1}{n}\right) \quad \forall n$; taking limits for $n \to +\infty$ we obtain: $|x| \leq \frac{\alpha^2}{\beta}$

3) $E = \mathrm{NL}$ or $E = \mathrm{PL}$, then it is trivial that $x\,\mathrm{Ne}\,E \iff x\,\mathrm{Ne}$ PL; therefore: $x\,\mathrm{Ne}\,E \iff \left|\frac{x}{y}\right| < \frac{\alpha}{\beta} \quad \forall y > \beta \iff |x| < \frac{\alpha y}{\beta} \quad \forall y > \beta \iff |x| \leq \alpha$

The proof of the last equivalence is analogous to the former one.

The definition of $E\,\mathrm{Ne}\,x$ is based on the same idea that $x\,\mathrm{Ne}\,E$:

Definition 3.2. Let x, E be a real number and a Q-label, respectively, then E *is negligible with respect to* x, $E\,\mathrm{Ne}\,x$, if

$$y\,\mathrm{Ne}\,x \quad \forall y \in E.$$

That is to say, $E\,\mathrm{Ne}\,x$ iff $x = 0$ and $E = [0]$, or, $x \neq 0$ and $\left|\frac{y}{x}\right| < \frac{\alpha}{\beta} \quad \forall y \in E$.

The possible cases for this relation are studied in the following:

Proposition 3.2.

1) If NL $\leq E$ or PL$\leq E$, then there does not exists any x such that $E\,\mathrm{Ne}\,x$.
2) If NL, PL $\not\leq E$, and $E \geq$ NM or PM, then $E\,\mathrm{Ne}\,x \iff |x| \geq \frac{\beta^2}{\alpha}$
3) If NL, PL, NM, PM $\not\leq E$, and E $\geq$ NS or PS, then $E\,\mathrm{Ne}\,x \iff |x| \geq \beta$
4) If $E = [0]$, then $E\,\mathrm{Ne}\,x\; \forall x \in R$

Proof:

1) If NL $\leq E$ or PL$\leq E$, then it is trivial that $E\,\mathrm{Ne}\,x \iff$ PL $\mathrm{Ne}\,x$; therefore: $E\,\mathrm{Ne}\,x \iff \left|\frac{y}{x}\right| < \frac{\alpha}{\beta} \quad \forall y > \beta \iff |x| > \frac{y}{\alpha}\beta \quad \forall y > \beta$: such x does not exist.
2) If NL, PL $\not\leq E$, and $E \geq$ NM or PM, then it is trivial that $E\,\mathrm{Ne}\,x \iff$ PM $\mathrm{Ne}\,x$; therefore: $E\,\mathrm{Ne}\,x \iff \left|\frac{y}{x}\right| < \frac{\alpha}{\beta} \quad \forall y \in (\alpha, \beta) \iff |x| > \frac{y}{\alpha}\beta \quad \forall y \in (\alpha, \beta) \iff |x| \geq \frac{\beta^2}{\alpha}$

   The last equivalence is proved in the following way:

   $\Longleftarrow$) trivial, because $y < \beta$

   $\Longrightarrow$) we have: $|x| > \frac{\beta(\beta - 1/n)}{\alpha}\, \forall n$; taking limits for $n \to +\infty$, we obtain $|x| \geq \frac{\beta^2}{\alpha}$.
3) If NL, PL, NM, PM $\not\leq E$, and E $\geq$ NS or PS, then it is trivial that: $E\,\mathrm{Ne}\,x \iff$ PS $\mathrm{Ne}\,x$; therefore: $E\,\mathrm{Ne}\,x \iff \left|\frac{y}{x}\right| < \frac{\alpha}{\beta} \quad \forall y \in (0, \alpha) \iff |x| > \frac{\beta y}{\alpha} \quad \forall y \in (0, \alpha) \iff |x| \geq \beta$

   The proof of the last equivalence is analogous to the former one.
4) $[0]\,\mathrm{Ne}\,x \iff 0\,\mathrm{Ne}\,x$, which is satisfied for any $x \in R$.

4 The Ne relation between labels

The negligibility relation between two qualitative labels $E_1 \operatorname{Ne} E_2$ expresses the negligibility between any pair of numbers representing E_1 and E_2:

Definition 4.1. Let E_1, E_2 be two Q-labels, then E_1 *is negligible with respect to* E_2, $E_1 \operatorname{Ne} E_2$, if

$$x \operatorname{Ne} y \quad \forall x, y \in R \,\text{such that}\; x \in E_1, y \in E_2.$$

That is to say, $E_1 \operatorname{Ne} E_2$ is equivalent to $x \operatorname{Ne} E_2$ $\forall x$ suchthat $x \in E_1$, or, equivalently, $E_1 \operatorname{Ne} y$ $\forall y$ such that $y \in E_2$.

The casuistry for this relation is given in the following:

Proposition 4.1.

1) $[0] \operatorname{Ne} E_2 \quad \forall E_2 \in S$
2) $\text{if}\, E_1 \neq [0]$, then $E_1 \operatorname{Ne} E_2 \iff E_1 \in \{\text{NS, PS, [NS,PS]}\}$, and $E_2 \in \{\text{NL,PL}\}$

Proof: it suffices to consider the case when E_2 is a positive basic label, because $E_1 \operatorname{Ne} E_2 \iff E_1 \operatorname{Ne} (-E_2)$, and, the fact that if $E_2 = B_1 \cup \ldots \cup B_k$, union of basic labels, then $E_1 \operatorname{Ne} E_2 \iff E_1 \operatorname{Ne} B_i \; \forall i$.

- $E_1 \operatorname{Ne} [0] \iff E_1 = [0]$
- E_1 Ne PS $\iff E_1 = [0]$, using proposition 3.1. 1)
- Using 3.1. 2): E_1 Ne PM $\iff |x| \leq \frac{\alpha^2}{\beta} < \alpha \quad \forall x \in E_1 \iff E_1 = [0]$
- Using 3.1. 3): E_1 Ne PL $\iff |x| \leq \alpha \quad \forall x \in E_1 \iff E_1 = [0]$ or $E_1 =$ PS or $E_1 =$ NS or $E_1 = [\text{PS}, \text{NS}]$

5 The Q − Ne relation between numbers and labels

The relations x Q − Ne E expresses the negligibility between x and some number which has E as Q-label:

Definition 5.1. Let x, E be a real number and a Q-label, respectively, then x *is Q-negligible with respect to* E, x Q − Ne E, if

$$x \operatorname{Ne} y \quad \text{for some}\; y \in E.$$

That is to say, x Q − NeE iff $E = [0]$ and $x = 0$, or, $E \neq [0]$ and $\exists y \in E - \{0\} : \left|\frac{x}{y}\right| < \frac{\alpha}{\beta}$.

So,the Q-Ne relation expresses the possibility of negligibility between x and E.

The possible cases for this relation are studied in the following:

Proposition 5.1.

1) If $E = [0]$, then x Q $-$ Ne E iff $x = 0$
2) If $E \geq$ NS or PS, and NL, PL, NM, PM $\not\leq E$, then x Q $-$ Ne E iff $|x| < \frac{\alpha^2}{\beta}$
3) If $E \geq$ NM or PM, and NL, PL $\not\leq E$, then x Q $-$ Ne E iff $|x| < \alpha$
4) If NL $\leq E$ or PL $\leq E$, then x Q $-$ Ne E $\forall x \in R$

Proof:

1) $x \text{ Q} - \text{Ne}\,[0] \iff x \,\text{Ne}\, 0 \iff x = 0$
2) If $E \geq$ NS or PS, and NL, PL, NM, PM $\not\leq E$, then it is trivial that x Q $-$ Ne $E \iff x$ Q $-$ Ne PS; therefore: x Q $-$ Ne $E \iff \exists y \in (0, \alpha) : \left|\frac{x}{y}\right| < \frac{\alpha}{\beta} \iff \exists y \in (0,\alpha) : |x| < \frac{\alpha y}{\beta} \iff |x| < \frac{\alpha^2}{\beta}$

The last equivalence is proved in the following way:

$\Longrightarrow$) $|x| < \frac{\alpha y}{\beta} < \frac{\alpha^2}{\beta}$

$\Longleftarrow$) if suffices to take y such that $\beta\frac{|x|}{\alpha} < y < \alpha$.

3) If $E \geq$ NM or PM, and NL, PL $\not\leq E$, then it is trivial that x Q $-$ Ne $E \iff x$ Q $-$ Ne PM; therefore: x Q $-$ Ne $E \iff \exists y \in (\alpha, \beta) : \left|\frac{x}{y}\right| < \frac{\alpha}{\beta} \iff \exists y \in (\alpha,\beta) : |x| < \frac{\alpha y}{\beta} \iff |x| < \alpha$

The proof of the last equivalence is analogous to the former one.

4) If NL $\leq E$ or PL $\leq E$, then it is trivial that x Q $-$ Ne $E \iff x$ Q $-$ Ne PL; therefore: x Q $-$ Ne $E \iff \exists y > \beta : \left|\frac{x}{y}\right| < \frac{\alpha}{\beta}$, which is satisfied for any $x \in R$.

The definition of E Q $-$ Ne x is based on the same idea that x Q $-$ Ne E:

Definition 5.2. Let x, E be a real number and a Q-label, respectively, then E *is Q-negligible with respect to* x, E Q $-$ Ne x, if

$$y \,\text{Ne}\, x \text{ for some } y \in E.$$

That is to say, E Q $-$ Ne x iff $E \geq [0]$ or $\exists y \in E : \left|\frac{y}{x}\right| < \frac{\alpha}{\beta}$.

So, the Q-Ne relation expresses the possibility of negligibility between E and x.

The casuistry for this relation is given in the following:

Proposition 5.2.

1) If $[0] \leq E$, then E Q $-$ Ne x $\forall x \in R$.
2) If $[0] \not\leq E$, and $E \geq$ NS or PS, then E Q $-$ Ne x $\forall x \in R - \{0\}$.
3) If $[0]$, NS, PS $\not\leq E$, and $E \geq$ NM or PM, then E Q $-$ Ne $x \iff |x| > \beta$
4) If $E =$ NL or PL, then E Q $-$ Ne $x \iff |x| > \frac{\beta^2}{\alpha}$

Proof:

1) If $[0] \leq E$, since $0 \,\mathrm{Ne}\, x$ is satisfied for any $x \in R$, then E Q − Ne $x \; \forall x \in R$
2) If $[0] \not\leq E$, and $E \geq$ NS or PS, then it is trivial that E Q − Ne $x \iff$ PS Q − Ne x; therefore:
   E Q − Ne $x \iff \exists y \in (0, \alpha) : \left|\frac{y}{x}\right| < \frac{\alpha}{\beta}$: satisfied for any $x \neq 0$.
3) If $[0]$, NS, PS $\not\leq E$, and $E \geq$ NM or PM, then it is trivial that E Q − Ne $x \iff$ PM Q − Ne x; therefore: E Q − Ne $x \iff \exists y \in (\alpha, \beta) : \left|\frac{y}{x}\right| < \frac{\alpha}{\beta} \iff \exists y \in (\alpha, \beta) : y < |x|\frac{\alpha}{\beta} \iff |x| > \beta$

   The last equivalence is proved in the following way:
   $\Longrightarrow$) $|x| > \frac{\beta y}{\alpha} > \beta$
   $\Longleftarrow$) it suffices to take y such that $\alpha < y < \min\{\beta, \frac{|x|\alpha}{\beta}\}$
4) If $E =$ NL or PL, then it is trivial that E Q − Ne $x \iff$ PL Q − Ne x; therefore: E Q − Ne $x \iff \exists y > \beta : \left|\frac{y}{x}\right| < \frac{\alpha}{\beta} \iff \exists y > \beta : y < \frac{|x|\alpha}{\beta} \iff |x| > \frac{\beta^2}{\alpha}$
   The proof of the last equivalence is analogous to the former one.

6 The Q − Ne relation between labels

The Q-negligibility relation between two qualitative labels $E_1 \,\mathrm{Ne}\, E_2$ expresses the possibility of negligibility between E_1 and E_2:

Definition 6.1. Let E_1, E_2 be two Q-labels, then E_1 *is Q-negligible with respect to* E_2, E_1 Q − Ne E_2, if

$$\exists x \in E_1, \exists y \in E_2 : x \,\mathrm{Ne}\, y.$$

That is to say,E_1 Q − Ne E_2 is equivalent to $\exists x \in E_1 : x$ Q − Ne E_2, or $\exists y \in E_2 : E_1$ Q − Ne y.

The casuistry for this relation is given in the following:

Proposition 6.1.

1) If $[0] \leq E_1$, then E_1 Q − Ne $E_2 \; \forall E_2$.
2) If $[0] \not\leq E_1$, and $E_1 \geq$ NS or PS, then E_1 Q − Ne $E_2 \; \forall E_2 \neq [0]$.
3) If $[0]$, NS, PS $\not\leq E_1$, then E_1 Q − Ne $E_2 \iff$ NL $\leq E_2$ or PL $\leq E_2$

Proof: it suffices to consider the case when E_2 is a positive basic label, because E_1 Q − Ne $E_2 \iff E_1 \,\mathrm{Ne}\, (-E_2)$, and, the fact that if $E_2 = B_1 \cup \ldots \cup B_k$, union of basic labels, then E_1 Q − Ne $E_2 \iff \exists B_i$ such that E_1 Q − Ne B_i.

- E_1 Q − Ne $[0] \iff \exists x \in E_1 : x \,\mathrm{Ne}\, 0 \iff \exists x \in E_1 : x = 0 \iff E_1 \geq [0]$
- E_1 Q − Ne PS $\iff \exists x \in E_1 : |x| < \frac{\alpha^2}{\beta} < \alpha \iff E_1 \geq$ PS or NS or $E_1 = [0]$
- E_1 Q − Ne PM $\iff \exists x \in E_1 : |x| < \alpha \iff E_1 \geq$ PS or NS or $E_1 = [0]$
- E_1 Q − Ne PL: any E_1 satisfy this inequality.

7 Example: a Heat Exchanger

Let us consider the example of the counter-current heat exchanger studied in [4], [2] and [8], [9], [10].

The variables considered for the heat exchanger are:

$DTH = T_{h_1} - T_{h_2}$ (temperature drop of the hot stream); $DTC = T_{c_1} - T_{c_2}$ (temperature rise of the cold stream); $DT1 = T_{h_1} - T_{c_1}$ (driving force at left end of the device); $DT2 = T_{h_2} - T_{c_2}$ (driving force at right end of the device); FH (molar-flowrate of the hot stream); KH (molar-heat of the hot stream); FC (molar-flowrate of the cold stream); KC (molar-heat of the cold stream).

The equations between the variables are:

$$\text{(e1)} \quad DTH - DT1 - DTC + DT2 = 0$$

$$\text{(e2)} \quad DTH \cdot KH \cdot FH = DTC \cdot KC \cdot FC$$

For this counter-current heat exchanger the order of magnitude relations considered in [2] and [8], [9] were:

$$\text{(r1)} \quad DT1 \text{ Ne } DTH$$

$$\text{(r2)} \quad KH \text{ Vo } KC$$

$$\text{(r3)} \quad DT1 \text{ Vo } DT2$$

Let us consider the case in which one disposes of some real quantitative data and only disposes of the qualitative descriptions of other ones.

For example, let us suppose that the real values of DTH and DTC can not be known, and only their qualitative labels are available. In this way, the relation (r1) will be replaced by:

$$\text{(r1}'\text{)} \quad DT1 \text{ Ne } [DTH]$$

Starting from these two equations and the relations (r1'), (r2) and (r3), the relation between [DTH] and [DTC] is studied using the inference rules presented in [8], [9] and the study of the relations introduced in this paper.

From definition 3.1., (r1') implies (r1), so, as proved in [8], [9]:

$$DTH \text{not Ne} \, DTC \quad \text{and} \quad DTC \text{not Ne} \, DTH$$

This implies, by definition 4.1.,

$$[DTH] \text{not Ne} \, [DTC] \quad \text{and} \quad [DTC] \text{not Ne} \, [DTH],$$

and, therefore, $[DTH]$ and $[DTC]$ can take any pair of values in S except (PS,PL) or (PL,PS).

So, it is impossible the case when the temperature drop of the hot stream is small and, at the same time, the temperature rise of the cold stream is large, which is a suitable condition for a heat exchanger. On the other hand, it is also impossible to have the optimal situation in which the temperature drop of the hot stream is large whereas the temperature of the cold stream when it is returned is nearly the same as before entering in the device.

In this case in which only the qualitative label of DTH can be known, let us start from table 5 of [San 96b] about the negligibility relation between DT2 and DTH and see how some improving is obtained:

a) Case $DT2 \leq DT1$:
We have $DT2 \,\mathrm{Ne}\, [DTH]$ because of (r1').
b) Case $DT2 > DT1$:
In this case, the following relation is obtained:

$$DT2 \; \mathrm{Q}-\mathrm{Ne}\, [DTH]$$

Proof:
b1. If $[DTH] \geq$ PS, then from (r1'), $DT1 < x \; \forall x \in [DTH]$, whence $DT1 = 0$, and by (r3), also $DT2 = 0$, and this is a contradiction with $DT2 > DT1$.
b2. Otherwise, let us suppose $[DTH] \not\geq$ PS. If $[DTH] \geq$ PL, then any x satisfies $x \; \mathrm{Q}-\mathrm{Ne}\, [DTH]$, and therefore, $DT2 \; \mathrm{Q}-\mathrm{Ne}\, [DTH]$. And if $[DTH] =$ PM, the assumption (r1') implies, as seen in section 3, that $DT1 \leq \frac{\alpha^2}{\beta} < \alpha$, so DT1 is small and also DT2 is small because (r3). By section 5, any small number is Q-negligible with respect to PM.

8 Conclusions

This paper represents a contribution to the modelization of some aspects of human reasoning such that the knowledge available is given by quantitative data on the one hand, and qualitative descriptions of other data, on the other hand.

It is presented a natural procedure for extending negligibility relation between two real quantities, to negligibility relations between real numbers and qualitative labels of the Absolute Orders of Magnitude Models. The system of inference rules introduced in [San96a] can be applied to the cases when data are given by relations between numbers and labels, and a new wider system of inference rules starting from the definitions given in this paper is now an on-going research.

References

1. Dague, P.: Order of Magnitude Revisited. In: 2nd International Workshop on Qualitative Phisics, Paris,(1988)
2. Dague,P.: Numeric Reasoning with Relative Orders of Magnitude. In: Proceedings of AAAI Conference, Washington, (1993)

3. Dague, P.: Symbolic Reasoning with Relative Orders of Magnitude.In: Proceedings of the 13th IJCAI, Chambéry, (1993)
4. Mavrovouniotis, M.L., Stephanopoulos,G.: Reasoning with Orders of Magnitude and Approximate Relations. In: Proceedings of AAAI Conference, Seatle (1987)
5. Mavrovouniotis, M.L., Stephanopoulos,G.: Formal Order of Magnitude Reasoning in Process Engineering. Computer Chem. Engng., Vol 12, n. 9/10, pp. 867-880, (1988)
6. Piera,N., Travé,L.: About Qualitative Equality: Axioms and Properties. 9th International Workshop on Expert System and their Applications, Avignon (1989)
7. Raiman,O.: Order of Magnitude Reasoning. In: Proceedings of the 5th National Conference on Artificial Intelligence (AAAI). Philadelphia, USA (1986)
8. Sánchez,M., Prats,F., Piera,N.: Mixing the Absolute and the Relative Order of Magnitude Reasoning in Qualitative Models. Research Report MA2-IT-96-0005, UPC, Barcelona (1996)
9. Sánchez,M., Prats,F., Piera,N.: Una formalización de relaciones de comparabilidad en modelos cualitativos. In: Boletín AEPIA n.6, pp 15,22, Madrid (1996)
10. Sánchez,M., Prats,F., Piera,N.: Degrees of negligibility in Qualitative Models. In: Proc. Journées Hispano-Françaises. Systèmes Intelligents et Contrôle Avancé. Barcelona (1996)
11. Travé,L., Piera,N.: The orders of Magnitude Models as Qualitative Algebras. In: Proc. 11th IJCAI, Detroit (1989)

Qualitative Reasoning under Uncertain Knowledge

M. Chachoua and D. Pacholczyk

LERIA, U.F.R Sciences 2, Boulevard Lavoisier
49045 ANGERS Cedex 01, France.
chachoua@univ-angers.fr, pacho@univ-angers.fr

Abstract. In this paper, we focus our attention on the processing of the uncertainty encountered in the common sense reasoning. Firstly, we explore the uncertainty concept and then we suggest a new approach which enables a representation of the uncertainty by using linguistic values. The originality of our approach is that it allows to reason on the uncertainty interval $[\![$*Certain, Totally uncertain*$]\!]$. The uncertainty scale that we use here, presents some advantages over other scales in the representation and in the management of the uncertainty. The axiomatic of our approach is inspired by the Shannon theory of entropy and built on the substrate of a symbolic many-valued logic.

1 Introduction

In the common sense reasoning, the uncertainty is usually expressed by using the linguistic expressions like *"very uncertain", "totally uncertain", "almost certain",...* The main feature of this uncertainty is its *qualitative* nature [10, 4]. So, several approaches have been proposed for the processing of this uncertainty category. Among these, we can quote the qualitative possibility theory [7] and the possibility logic [8], the qualitative evidence theory [17] and some qualitative probabilities theories [19, 9, 21, 2, 3, 15, 6, 18, 14]. These approaches offer better formalism for uncertainty representation of the common-sense knowledge. Nevertheless, in the natural language, the human subject uses generally two forms to express his uncertainty [12]: (1) *explicitly*, as for example in the statement "*I am totally uncertain of my future*" where, the term "*totally uncertain*" expresses an *ignorance degree* and (2) *implicitly*, very often in the *belief* form as for example in the statement "*It is very probable that Tweety is a penguin*". The term "*very probable*" designates a *belief degree* [12]. However, all approaches quoted previously concern rather the uncertainty expressed under belief form.

In this paper we suggest a new approach [1] whose objective is to contribute mainly to the qualitative processing of uncertainty expressed in the *ignorance* form. But it can also process the belief form. For example we can translate the statement "*It is very probable that Tweety is a penguin*" as "*It is almost certain (i.e. little ignorance) that Tweety is a penguin*", whereas, a statement expressed

[1] The first versions of this have been presented in [4, 5].

in the ignorance form can not always be translated under the belief form. This is the case of the statement "*I am totally uncertain of my future*".

In this approach, the uncertainty is represented by using the linguistic values in the interval [*Certain, Totally uncertain*]. This graduation scale presents at least two advantages. The first one concerns the representation of the ignorance situation that the scales used in other qualitative approaches do not allow. The second advantage concerns the management of the uncertainty. Indeed, in the qualitative management of uncertain knowledge, one often leads to intervals of *belief*, *certainty...* Then the question is to choose one value from these intervals. In our approach, to palliate this problem, we can choose the greatest value of an interval. The choice corresponds to the principle of maximum entropy [11].

In section 2, we explore the uncertainty concept and we show the difference between ignorance form and belief form of the uncertainty concept.

Besides, the uncertainty concept is *gradual.* So, to take account of this feature, our approach is built on the substrate of a *many-valued logic* suggested by Pacholczyk [15]. This logic will be presented in section 3.

In section 4, we discuss our method of uncertainty representation. This method consists of the definition in the logical language of a many-valued predicate called *Uncert.* This predicate satisfies a set of axioms which governs the uncertainty concept, that we present in section 5. Thanks to this, in section 6 we obtain some theorems and particularly new generalization rule of Modus Ponens. These properties offer a formal framework for the qualitative management of the uncertainty. In section 7, we presents the mains differences between our theory and others qualitative approaches of uncertainty. Finally, in section 8, we presents an application example.

2 Uncertainty concept

Generally, in the universe of information, knowledge is considered to be *certain* if it is *either true* or *false.* Otherwise, it is considered as *uncertain.* Let us consider the following example.

Example 1. Let A, B, C, D, E, and F be six towns such as [4]:
1. *All* the inhabitants of the town A always say the truth,
2. *All* the inhabitants of the town B are liars,
3. *A minority* of inhabitants of the town C always says the truth,
4. *A majority* of inhabitants of the town D always says the truth,
5. *Half* of inhabitants of town F are liars.

Supposing we don't have any knowledge about the town E. let us designate by $H(X)$ an ordinary inhabitant from the town X ($X \in \{A, B, C, D, E, F\}$) and let us assume that we will be interested by the truth value of the sentence: "*H(X) is a liar*".

It is clear that the sentence "*$H(A)$ is a liar*" is *false* and the sentence "*$H(B)$ is a liar*" is *true.* So these two sentences are **certain**. In probability terms (or belief), we have Prob(*"H(A) is a liar"*)=0 and Prob(*"H(B) is a liar"*)= 1.

Nevertheless, the determination of the truth value of some sentences like "*$H(X)$ is a liar*" with $X \in \{C, D, E, F\}$ is impossible with the available knowledge. Thus, these sentences are **uncertain**. However, note that the ignorance is maximal about inhabitants of the towns E and F. Nevertheless the probability (belief) of "*$H(E)$ is a liar*" is not *estimable.* Indeed, we can not attribute Prob($H(E)$ *is a liar*) = Prob($H(E)$ *is not a liar*) $\approx \frac{1}{2}$ as in the town F, because in this town there is no ignorance about quantity of liars as in the town E. A study on the difference between these situations is presented in [8]. Note again that, intuitively, the ignorance about the inhabitants of the towns C and D are approximately equal, but their probabilities can be very different.

It results that one of the main features of the uncertainty concept is the ignorance of the truth values. According to Shannon's entropy theory [20], the uncertainty concept refers also to the information deficiency. Indeed, Shannon has shown that a measure of *information* can also be used for measuring *ignorance* [20, 13]. It follows that *the ignorance* degree expresses the degree of the information deficiency to determine the truth value. However, *the belief degree* refers rather to the information available.

In the natural language, to evaluate uncertainty, the human subject refers to a set of adverbial expressions like *almost certain, very uncertain...* This set allows him to build a subjective scale of uncertainty degrees. In our approach we reproduce the same method. So, in the following section we introduce the algebraic structures on which our method will be constructed.

3 Algebraic structures

The algebraic structures and the many-valued logic that we present here have been already presented by Pacholczyk [15, 16].

3.1 Chains of De Morgan

Let $M \geq 2$ be an integer. Let us designate by $\mathcal{W}$ the interval $[1, M]$ completely ordered by the relation "$\leq$" , and by "n" the application such that $n(\beta) = M + 1 - \beta$. In these conditions $\{ \mathcal{W}, \wedge, \vee, n\}$ is a lattice of De Morgan with: $\alpha \wedge \beta = min(\alpha, \beta)$ and $\alpha \vee \beta = max(\alpha, \beta)$.
Let $\mathcal{L}_M = \{\tau_1, \tau_2, ..., \tau_M\}$ be a set where: $\alpha \leq \beta \Leftrightarrow \tau_\alpha \leq \tau_\beta$. Thus, $\{\mathcal{L}_M, \leq\}$ is a chain in which the least element is τ_1 and the greatest element is τ_M.
We define in $\mathcal{L}_M$, two operators "$\wedge$" and "$\vee$" and a decreasing involution "$\sim$" by the relation: $\tau_\alpha \wedge \tau_\beta = \tau_{min(\alpha,\beta)}$, $\tau_\alpha \vee \tau_\beta = \tau_{max(\alpha,\beta)}$, and $\sim \tau_\alpha = \tau_{n(\alpha)}$.
Likewise, operators v_γ ($\gamma \in \mathcal{W}$) are defined as $\mathcal{L}_M$ in the following way:
If $a = b$ then $v_a\tau_b = \tau_M$ else $v_a\tau_b = \tau_1$.

Remark. In the context of the many-valued logic used here, the qualitative values v_α and τ_α are associated by the relation:

$$\text{“}x \ is \ v_\alpha \ A\text{''} \iff x \ is \ v_\alpha \ A \iff \text{“}x \ is \ A\text{''} \ is \ \tau_\alpha - true$$

where x and A designates respectively an object and a concept. Note that these equivalences can be viewed as generalizations of Tarski criteria [15, 16].

Example 2. For $M = 5$, one could introduce one possible set of linguistic degrees totally ordered like: $\mathcal{L}_5 = \{$*not-at-all, little, enough, very, totally*$\}$.

So, in these conditions, if the assertion "*Tom is very young* " is *true*, then it is equivalent to "*Tom is young* " is *very-true.*
In the following, the chain $\{\mathcal{L}_M, \leq, \wedge, \vee, \sim\}$ will be used as the support of the representation of truth degrees.

3.2 Interpretation and satisfaction of formulae

The formal system of many-valued logic used here is can be found in [15].

Definition 1. *We call an interpretation structure $\mathcal{A}$ of the language $\mathcal{L}$, the pair $< \mathcal{D}, \mathcal{R}_n >$ for $n \in N$, where $\mathcal{D}$ is a non-empty set called domain of $\mathcal{A}$ and $\mathcal{R}_n$ a multi-set*[2] *in $\mathcal{A}$.*

Definition 2. *Let $\mathcal{V} = \{z_1, z_2, .., z_n, ..\}$ be the infinite countable set of individual variables of the formal system. We call a* valuation of variables, *a sequence denoted $x =< x_0, ..., x_n, ... >$ where $\forall i, x_i \in \mathcal{D}$. So, if x is a valuation, then $x(n/a) =< x_0, ..., x_{n-1}, a, x_{n+1}, ... >$.*

Definition 3. *Let Φ be a formula of $\mathcal{SF}$*[3] *and x a valuation. The relation "x* satisfies Φ to a degree v_α in $\mathcal{A}$" *denoted $\mathcal{A} \models^x_\alpha \Phi (\alpha \in \mathcal{W})$ is defined by:*

1. $\mathcal{A} \models^x_\alpha \mathcal{P}_n(z_{i_1}, ..., z_{i_k}) \Longleftrightarrow < z_{i_1}, ..., z_{i_k} > \in_\alpha \mathcal{R}_n$

2. $\mathcal{A} \models^x_\alpha \neg\Phi \Longleftrightarrow \mathcal{A} \models^x_\beta \Phi$, *with* $\tau_\beta = \sim \tau_\alpha$

3. $\mathcal{A} \models^x_M V_\alpha\Phi \Longleftrightarrow \mathcal{A} \models^x_\alpha \Phi$

4. $\mathcal{A} \models^x_\alpha \Phi \cup \Psi \Longleftrightarrow \mathcal{A} \models^x_\beta \Phi$ *and* $\mathcal{A} \models^x_\gamma \Psi$ *with* $\tau_\gamma \vee \tau_\beta = \tau_\alpha$

5. $\mathcal{A} \models^x_\alpha \Phi \cap \Psi \Longleftrightarrow \mathcal{A} \models^x_\beta \Phi$ *and* $\mathcal{A} \models^x_\gamma \Psi$ *with* $\tau_\gamma \wedge \tau_\beta = \tau_\alpha$

6. $\mathcal{A} \models^x_\alpha \Phi \supset \Psi \Longleftrightarrow \mathcal{A} \models^x_\beta \Phi$ *and* $\mathcal{A} \models^x_\alpha \Psi$ *with* $\tau_\beta \rightarrow \tau_\gamma = \tau_\alpha$

7. $\mathcal{A} \models^x_\alpha \exists z_n\Phi \Longleftrightarrow \tau_\alpha = Max\ \{\tau_\gamma \mid \mathcal{A} \models^{x(n/a)}_\gamma \Phi, a \in \mathcal{D}\}$

8. $\mathcal{A} \models^x_\alpha \forall z_n\Phi \Longleftrightarrow \tau_\alpha = Min\ \{\tau_\gamma \mid \mathcal{A} \models^{x(n/a)}_\gamma \Phi, a \in \mathcal{D}\}$

Definition 4. *Let ϕ be a formula. We say that ϕ* is v_α -true *in $\mathcal{A}$ if and only if we have a valuation x such that x v_α* -satisfies *ϕ in $\mathcal{A}$.*

[2] The notion of multi-set was introduced by M. De Glas in 1988 (see [15] and [1]). In this theory $x \in_\alpha \mathcal{R}_n \Longleftrightarrow x$ belongs to degree τ_α to the multi-set $\mathcal{R}_n$. Note that this multi-set theory is an axiomatics approach to the fuzzy sets theory of Zadeh. In this theory $x \in_\alpha A$ in the formal representation of $\mu_A(x) = \alpha$.

[3] $\mathcal{SF}$ designates the formulae set for the system of many-valued predicates calculus.

4 Uncertainty representation

Let $\mathcal{L}$ be a first order language, $\mathcal{A}$ an interpretation structure of $\mathcal{L}$ and Ω a set of formulae of $\mathcal{SF}$ such that:

$$\Omega = \{\phi \in \mathcal{SF}, \mathcal{A} \models^{x}_{M} \phi \ or\ \mathcal{A} \models^{x}_{1} \phi\}$$

In other terms, for a valuation x, all formulae of Ω are either *satisfied*, or *not satisfied* in $\mathcal{A}$ (ie. true or false in $\mathcal{A}$). The uncertainty of a formula ϕ of Ω is treated thanks to a particular many-valued predicate called *Uncert* which has been added in the first order many-valued predicate logic.

Definition 5. The predicate *Uncert* is defined as follows:

▷ Uncert is a many-valued predicate such that:

$$\forall \varphi \in \Omega \text{ Uncert } (\varphi) \in \mathcal{SF}.$$

▷ Given an interpretation $\mathcal{A}$ of $\mathcal{L}$, for all formula of Ω one has:

$$\mathcal{A} \models^{x(0/\phi)}_{\gamma} Uncert(V_{\chi}\varphi) \iff \text{Uncert}(V_{\chi}\phi) \text{ is } v_{\gamma} - \textit{true} \text{ in } \mathcal{A}$$
$$\iff \text{“}\phi \text{ is } t_{\chi} - \textit{true}\text{” is } u_{\gamma}\textit{-uncertain} \text{ in } \mathcal{A}$$

with $\tau_{\gamma} \in [\tau_1, \tau_M]$ and $\tau_{\chi} \in \{\tau_1, \tau_M\}$.

Notations: If there is no confusion, in the following, we use:
1. $\mathcal{A} \models_{\gamma} Uncert(V_{\chi}\phi)$ instead of $\mathcal{A} \models^{x(0/\phi)}_{\gamma} Uncert(V_{\chi}\varphi)$.
2. $\mathcal{A} \models_{\gamma} Uncert(\phi)$ instead of $\mathcal{A} \models_{\gamma} Uncert(V_M\phi)$.

Example 3. By choosing $M = 5$, one could introduce the following ordered set of linguistic degrees.
D_{t_5}: {*not-at-all-true, little-true, true-enough, very-true, totally-true*} which corresponds to $v_{\alpha} - \textit{true}$ with $\alpha = 1, 2, ..., 5$.
D_{u_5}: {*not-at-all-uncertain, little-uncertain, uncertain-enough, very-uncertain, totally-uncertain*} which corresponds to $u_{\alpha} - \textit{uncertain}$ with $\alpha = 1, 2, ..., 5$.

Besides, in the natural language, one normally uses “*certain*” and “*almost certain*” rather than “*not at all uncertain*” and “*little-uncertain*”. So, with these terms we obtain the uncertainty scale[4] represented by figure 1.

Certain	Almost-certain	Uncertain-enough	Very-uncertain	Totally-uncertain

Fig.1: Uncertainty scale

So, if Uncert($V_M\phi$) is *very true*, then *“ϕ is true”* is *very uncertain.*
Besides, to manage rationally the uncertainty, it is necessary that the predicate *“Uncert”* satisfy an axioms set which governs this concept.

[4] Please note that this scale and the linguistic terms are chosen subjectively by a human expert.

5 Axioms of *Uncert*

According to the sense that we have given in the previous section, a formula is *certain* if its *truth value*[5] is known. From this, we derive the first axioms:

Axiom 5.1 $\forall\phi \in \Omega, \mathcal{A} \models_M \phi \Longrightarrow \mathcal{A} \models_1 \textbf{Uncert}(V_M\phi)$.

Axiom 5.2 $\forall\phi \in \Omega, \mathcal{A} \models_1 \phi \Longrightarrow \mathcal{A} \models_1 \textbf{Uncert}(V_1\phi)$.

Besides, two equivalent formulae have the same truth value. By this fact, their uncertainty is the same. So, the third axiom is:

Axiom 5.3 $\forall\phi, \varphi \in \Omega, \mathcal{A} \models_M (\phi \equiv \varphi) \Rightarrow \{\mathcal{A} \models_\alpha \textbf{Uncert}(\phi) \Leftrightarrow \mathcal{A} \models_\alpha \textbf{Uncert}(\varphi)\}$.

The formulae ϕ and $\neg\phi$ are mutually exclusive. However, if the formulae ϕ and $\neg\phi$ are uncertain, then considering the available information, one will choose the most relevant formula (ie. the most certain formula). Thus, the fourth axiom is:

Axiom 5.4 $\forall\phi \in \Omega$, **if** $\{\mathcal{A} \models_\alpha \textbf{Uncert}(\phi)$ **and** $\mathcal{A} \models_\beta \textbf{Uncert}(\neg\phi)\}$ **then** $\mathcal{A} \models_{\alpha\wedge\beta} \textbf{Uncert}(V_\chi\phi)$ **with** $\{$**if** $\tau_\alpha \leq \tau_\beta$ **then** $\tau_\chi = \tau_M$, **else** $\tau_\chi = \tau_1\}$

Let us consider now two non contradictory formulae Ψ and Φ. Knowing the uncertainties on Ψ and Φ, what will be the uncertainty on their conjunction?

If Ψ is true (thus no uncertainty on Ψ) the determination of the truth value of $(\Psi \cap \Phi)$ will depend only on that of Φ. In these terms, the uncertainty on $(\Psi \cap \Phi)$ is equal to the uncertainty on Φ. It follows that, if one of the formulae is *totally uncertain*, their conjunction is also *totally uncertain*. So, the resulting uncertainty from $(\Psi \cap \Phi)$ is, according to Shannon theory of entropy [20], equal to the *sum* individual uncertainties of Φ and Ψ. In the qualitative domain, we use this same property. So, instead the numerical operator "+", we use a qualitative additive operator S defined bellow. Thus we have the last axiom is then:

Axiom 5.5 $\forall\Phi, \Psi \in \Omega$, **if** $\Psi \not\equiv \neg\Phi$ **and** $\{\mathcal{A} \models_\alpha \textbf{Uncert}(\Phi), \mathcal{A} \models_\beta \textbf{Uncert}(\Psi)\}$ $\Longrightarrow \mathcal{A} \models_\gamma \textbf{Uncert}(\Psi \cap \Phi)$ **with** $\tau_\gamma = S(\tau_\alpha, \tau_\beta)$.

Definition 6. *The operator S is defined in order to satisfy the following properties of classical sum:* $\forall(\tau_\alpha, \tau_\beta, \tau_\gamma, \tau_\delta) \in [\tau_1, \tau_M]$,
s_1. $S(\tau_\alpha, \tau_1) = \tau_\alpha$,
s_2. $S(\tau_\alpha, \tau_\beta) = S(\tau_\beta, \tau_\alpha)$,
s_3. $S(\tau_\alpha, S(\tau_\beta, \tau_\gamma)) = S(S(\tau_\alpha, \tau_\beta), \tau_\gamma)$
s_4. $\{\tau_\alpha \leq \tau_\beta$ *and* $\tau_\gamma \leq \tau_\delta\} \Longrightarrow S(\tau_\alpha, \tau_\gamma) \leq S(\tau_\beta \leq \tau_\delta)$.

6 Uncertainty management

In this section we present and prove the fundamental theorems regarding the management of uncertainty.

[5] The truth values "*true*" and " *false*" correspond respectively to τ_M and τ_1

Theorem 7. *Uncertainty of the conjunction of several formulae.*
$\forall \Phi_i \in \Omega, i \in \{1,...,n\}$, **if the formulae** $\Phi_{i=1,..,n}$ **are non contradictory and for all** i **we have** Φ_i **is** u_{β_i}**-uncertain, then the conjunction** $(\Phi_1 \cap ... \cap \Phi_n)$ **is** u_χ**-uncertain with** $\tau_\chi = S(\tau_{\beta_1}, ...S(\tau_{\beta_{n-1}}, \tau_{\beta_n})...)$.

Proof. Consequence of the axiom 5.5 □

Corollary 8. *If* $\forall \Phi_i \in \Omega, i \in \{1,...,n\}$, $\mathcal{A} \models_1 Uncert(\Phi_{i \neq a})$ *and* $\mathcal{A} \models_\alpha Uncert(\Phi_a), (a \in \{1,..,n\})$. *Using the theorem 7 one obtains:* $\mathcal{A} \models_\chi Uncert(\Phi_1 \cap ... \cap \Phi_a \cap ... \cap \Phi_n)$ *with* $\tau_\chi = \tau_\alpha$.

In other terms, the conjunction of several formulae is u_α*-uncertain if none of them is less uncertain than* u_α*-uncertain.*

Theorem 9. *Uncertainty of the disjunction of two formulae.*
$\forall \phi \in \Omega, \forall \varphi \in \Omega$, **for an interpretation** $\mathcal{A}$, **if we have:**
φ **is** u_α**-uncertain** *and* Φ **is** u_β**-uncertain,**
then $(\varphi \cup \phi)$ **is** u_γ**-uncertain, with** $\tau_\gamma \leq \tau_\alpha \wedge \tau_\beta$.

Proof. $\forall \phi \in \Omega, \forall \varphi \in \Omega$, we have:
(1) $\mathcal{A} \models_M (\phi \equiv (\phi \cup \neg\varphi) \cap (\phi \cup \varphi))$ and
(2) $\mathcal{A} \models_M (\varphi \equiv (\varphi \cup \neg\phi) \cap (\phi \cup \varphi))$
Let suppose that: $\mathcal{A} \models_\alpha \text{Uncert}(\varphi)$, $\mathcal{A} \models_\beta \text{Uncert}(\phi), \mathcal{A} \models_\gamma \text{Uncert}(\varphi \cup \phi)$, $\mathcal{A} \models_{\sigma_1} \text{Uncert}(\phi \cup \neg\varphi), \mathcal{A} \models_{\sigma_2} \text{Uncert}(\varphi \cup \neg\phi)$.
Using the axioms 5.3,5.5 and the equations (1) and (2) one obtains:
$\tau_\alpha = (\tau_{\sigma_1}, \tau_\gamma)$ and $\tau_\beta = (\tau_{\sigma_2}, \tau_\gamma) \Rightarrow \tau_\gamma \leq \tau_\alpha \wedge \tau_\beta$ □

Theorem 10. *Uncertainty of the disjunction of several formulae .*
$\forall \Phi_i \in \Omega, i \in \{1,...,n\}$ **if we have several formulae** $\Phi_{i=1,..,n}$ **such that, for all** i, Φ_i **is** u_{β_i}**-uncertain, then the disjunction** $(\Phi_1 \cup ... \cup \Phi_n)$ **is** u_χ**-uncertain with** $\tau_\chi \leq \tau_{\beta_1} \wedge ... \wedge \tau_{\beta_n}$.

Proof. Consequence of the theorem 9 □

Corollary 11. $\forall \Phi_{i=1,...,n} \in \Omega$, *if* $\exists \Phi_a \mid \Phi_a \in \Omega$, *and* $\mathcal{A} \models_1 Uncert(\Phi_a)$, *then using the theorem 10 one has:* $\mathcal{A} \models_\chi Uncert(\Phi_1 \cup \Phi_a \cup ... \cup \Phi_n)$ *with* $\tau_\chi = \tau_1$
So, the disjunction of formulae is certain *if only one of them is* certain.

Theorem 12. *Generalized Modus Ponens rule.*
$\forall \phi \in \Omega, \forall \varphi \in \Omega$, **for an interpretation** $\mathcal{A}$, **if we have:**
φ **is** u_α**-uncertain** *and* $(\varphi \supset \phi)$ **is** u_β**-uncertain,**
then ϕ **is** u_λ**-uncertain with** $\tau_\lambda \leq S(\tau_\alpha, \tau_\beta)$.

Proof. $\forall \phi \in \Omega, \forall \varphi \in \Omega$, $\mathcal{A} \models_M ((\phi \cap \varphi) \equiv ((\varphi \supset \phi) \cap \varphi))$
According to the axiom 5.3 if $\mathcal{A} \models_\gamma \text{Uncert}(\phi \cap \varphi)$
then $\mathcal{A} \models_\gamma \text{Uncert}((\varphi \supset \phi) \cap \varphi))$. Using the axiom 5.5 one can write:
$\mathcal{A} \models_\gamma \text{Uncert}((\varphi \supset \phi) \cap \varphi) \Rightarrow \mathcal{A} \models_\beta \text{Uncert}(\varphi \supset \phi)$ and $\mathcal{A} \models_\alpha \text{Uncert}(\varphi)$ such as: $\tau_\gamma = S(\tau_\beta, \tau_\alpha)$. Always according the axiom 5.5:
$\forall \phi \in \Omega, \forall \varphi \in \Omega$, if $\mathcal{A} \models_\lambda \text{Uncert}(\phi)$, then $\tau_\lambda \leq \tau_\gamma$. Finally $\tau_\lambda \leq S(\tau_\alpha, \tau_\beta)$ □

Theorem 13. *Combination of uncertainties*
$\forall\phi \in \Omega, \forall\varphi \in \Omega$, for an interpretation $\mathcal{A}$, if we have:
ϕ is u_α-uncertain, and $(\phi \supset \Phi)$ is u_β-uncertain (1)
φ is u_γ-uncertain and $(\varphi \supset \Phi)$ is u_σ-uncertain (2)
Then Φ is u_μ-uncertain with $\tau_\mu \leq S(\tau_\alpha, \tau_\beta) \wedge S(\tau_\gamma, \tau_\sigma)$.

Proof. Using Modus Ponens and the equations (1) and (2) one obtains:
$\mathcal{A} \models_\mu \text{Uncert}(\Phi)$ with $\tau_\mu \leq S(\tau_\alpha, \tau_\beta)$ and $\tau_\mu \leq S(\tau_\gamma, \tau_\sigma)$.
Therefore $\tau_\mu \leq S(\tau_\alpha, \tau_\beta) \wedge S(\tau_\gamma, \tau_\sigma)$ □

Corollary 14. *$\forall\phi \in \Omega, \forall\varphi \in \Omega, \forall\Phi \in \Omega$ if we have:*
$\mathcal{A} \models_\alpha \mathit{Uncert}(\phi)$, and $\mathcal{A} \models_\beta \mathit{Uncert}(\phi \supset \Phi)$ (1)
$\mathcal{A} \models_\gamma \mathit{Uncert}(\varphi)$ and $\mathcal{A} \models_\sigma \mathit{Uncert}\ (\varphi \supset \neg\Phi)$ (2)
Then, using the Modus Ponens, the equations (1) and (2) give respectively:
$\mathcal{A} \models_\mu \mathit{Uncert}(\Phi)$ and $\mathcal{A} \models_\nu \mathit{Uncert}(\neg\Phi)$ with $\tau_\mu \leq S(\tau_\alpha, \tau_\beta)$ and $\tau_\nu \leq S(\tau_\gamma, \tau_\sigma)$.
According the axiom 5.4, one obtains: $\mathcal{A} \models_\theta \mathit{Uncert}(V_\chi\Phi)$ with
$\tau_\theta \leq S(\tau_\gamma, \tau_\sigma) \wedge S(\tau_\alpha, \tau_\beta)$ and $\tau_\chi = \tau_M$ if $S(\tau_\alpha, \tau_\beta) \leq S(\tau_\gamma, \tau_\sigma)$ else $\tau_\chi = \tau_1$.

Theorem 15. *Propagation of uncertainty.*
$\forall(\phi, \varphi, \Phi) \in \Omega$ for an interpretation $\mathcal{A}$, if $(\phi \supset \Phi)$ is u_β-uncertain *and* **$(\Phi \supset \varphi)$ is u_σ-uncertain then $(\phi \supset \varphi)$ is u_μ-uncertain with $\tau_\mu \leq S(\tau_\beta, \tau_\sigma)$.**

Proof. $\forall\phi, \varphi, \Phi \in \Omega$ if $\mathcal{A} \models_\beta \text{Uncert}(\phi \supset \Phi)$ and $\mathcal{A} \models_\sigma \text{Uncert}(\Phi \supset \varphi)$
Then according to the axiom 5.5 we have: $\mathcal{A} \models_\delta \text{Uncert}((\phi \supset \Phi) \cap (\Phi \supset \varphi))$ with $\tau_\delta = S(\tau_\beta, \tau_\sigma)$ and $\mathcal{A} \models_M (((\phi \supset \Phi) \cap (\Phi \supset \varphi)) \supset (\phi \supset \varphi))$
$\Rightarrow \mathcal{A} \models_1 \text{Uncert}(((\phi \supset \Phi) \cap (\Phi \supset \varphi)) \supset (\phi \supset \varphi))$. Using the Modus Ponens, one has: $\mathcal{A} \models_\mu \text{Uncert}(\phi \supset \varphi)$ with $\tau_\mu \leq S(\tau_\delta, \tau_1) = \tau_\delta$. Finally, $\tau_\mu \leq S(\tau_\beta, \tau_\sigma)$ □

6.1 Strategy

In the previous theorems, by using the maximum entropy principle [11], one can replace the sign "$\leq$" by "$=$" . As Jaynes suggests it, this choice is a kind of assurance which protects against the false predictions. This principle enables us to palliate the difficulty of the management of the uncertainties intervals.

7 Some differences with other qualitative approaches

The axiomatic of our approach is built to process the uncertainty expressed under ignorance form. However qualitative probabilities approaches [19, 9, 21, 2, 15, 6, 18, 14] and the qualitative approach of the evidence theory [17] allow rather to process especially the expressed uncertainty in the belief form. So, in uncertainty representation point of view, our approach presents the advantage to represent explicitly the situation of ignorance such that it is expressed in the natural language. In the uncertainty management, one can make emerge main two differences between our approach and others qualitative approaches. The first concerns the implication (or the inclusion in set terms). So, if A and B are two formulae of Ω, one has: $A \supset B \Longrightarrow C(A) \leq C(B)$, where $C()$ can designate a measure of, probability, possibility, necessity or credibility. However, in our approach one has: $A \supset B \Longrightarrow I(A) \geq I(B)$, where $I()$ designate a mesure of ignorance (ie the truth degree of the formulae Uncert(A) and Uncert(B)).

The second difference is relative to the conjunction of formulae (or intersection in set terms). So, generally in all qualitative approaches quoted in the former, one has: $C(A \cap B) \approx C(A) \otimes C(B)$, where $\otimes$ designates a multiplicative qualitative operator. In our approach one has: $I(A \cap B) = I(A) \oplus I(B)$, where $\oplus$ designates an qualitative additive operator. As we have underlined it, the qualitative processing of the uncertainty ends very often to intervals of uncertainties. However, when the uncertainty is represented in the belief form (propability, credibility...), the exploitation of these intervals is problematical. Thus, when the uncertainty is represented in the ignorance form, one can choose the maximal value of the interval.

8 Application

In this example, we will use the uncertainty scale defined in the example 3 and in order to lighten the notation, in the following example we will use:
Uncert(Φ) $\equiv u_\beta -$ *uncertain* instead of $\mathcal{A} \models_\beta$Uncert($\Phi$)

Example 4. Let us assume that we have the following knowledge:
1. Uncert((Tweety is a bird) $\supset$ (Tweety fly)) $\equiv$ *almost certain.*
2. Uncert((Tweety is a Penguin) $\supset$ (Tweety is a bird)) $\equiv$ *certain.*
3. Uncert((Tweety is a Penguin) $\supset$ (Tweety doesn't fly)) $\equiv$ *certain.*
4. Uncert(Tweety fly) $\equiv$ *totally uncertain.*

Let us assume now that we have obtained the following information:

5. Uncert(Tweety is a Penguin) $\equiv$ *certain.*

By using the Modus Ponens with the knowledge (5), (2) and (3), we obtain:
d_1 Uncert(Tweety is a bird) $\equiv$ *certain.*
d_2 Uncert(Tweety doesn't fly) $\equiv$ *certain.*
The knowledge [d_1] and (1) gives by using the Modus Ponens rule:
d_3 Uncert(Tweety fly) $\equiv$ *almost certain.*
Using the corollary 14 and the knowledge [d_2] and [d_3], finally we obtain:
d_4 Uncert(Tweety doesn't fly)$\equiv$ *certain.*

Thus the conclusion is: It is *certain* that Tweety doesn't fly.

9 Conclusion

In this paper we were interested to the processing of uncertainty encoded into a qualitative way. We have explored the uncertainty concept and we have presented the last results obtained with a new qualitative approach of uncertainty processing. This approach offers the possibility to represent explicitly the uncertainty, as it could be evaluated subjectively, by using linguistic values. The graduation scale used here allows us to handle several special cases of uncertainty, including the situation of total ignorance.

References

1. H. Akdag, M. De-Glas, and D. Pacholczyk. A qualitative theory of uncertainty. *Fundamenta Informaticae*, 17(4):333–362, 1992.
2. R. Aleliunas. A new normative theory of probabilistic logic. In *Procceedings of CSCSI'88*, pages 67–74, 1988.
3. F. Bacchus. Lp, a logic for representing and reasoning with statistical knowledge. *Computational Intelligence*, 6:209–231, 1990.
4. M. Chachoua and D. Pacholczyk. Symbolic processing of the uncertainty of common-sense reasoning. In *International Conference on Knowledge Based Computer Systems KBCS'96*, pages 217–228, Juhu, Bombay, December 1996.
5. M. Chachoua and D. Pacholczyk. Qualitative reasoning under uncertainty. In *International Conference FLAIRS'98*, page (To appear), Florida,USA, 1998.
6. A. Y. Darwiche and M. L. Ginsberg. A symbolic generalization of probability theory. In *Proceedings of the American Association for Artificial Intelligence*, pages 622–627, 1992.
7. D. Dubois. Belief structure, possibility theory, decomposable confidence measures on finite sets. *Computer and Artificial Intelligence*, 5(5):403–417, 1986.
8. D. Dubois and H. Prade. Modeling uncertainty and vague knowledge in possibility and evidence theories. In L.N. Kanal R.D. Shachter, TS. Levtt and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence 4*, volume 9, pages 303–318. Elsevier Science Publishers B.V, 1990.
9. P. Gärdenfors. Qualitative probability as an intensional logic. *Philosophical Logic*, 4:177–185, 1975.
10. D. Hernändez. *Qualitative representation of spatial knowledge.* Springer Verlag Edition, 1994.
11. E.T. Jaynes. On the rationale of maximum entropy methods. *IEEE*, 70(9):939–952, 1982.
12. E. Kant. *Logique.* Librairie Philosophique J. Vrin,, Paris, 1966.
13. G. J. Klir and T. A. Folger. *Fuzzy sets, uncertainty and information.* Prentice-all international Edition, 1988.
14. D. Lehmann. Generalized qualitative probability: Savage revisited. In *Proceedings of the Twelfth Annual Conference on Uncertainty in Artificial Intelligence (UAI-96)*, pages 381–388, Portland, Oregon, 1996.
15. D. Pacholczyk. *Contribution au traitement logico-symbolique de la connaissance, Thèse d'état.* Universitè Pièrre et Marie Curie, Paris 6, Avril 1992.
16. D. Pacholczyk. A logico-symbolic probability theory for the management of uncertainty. *CCAI*, 11(4):3–70, 1994.
17. S.D. Parsons and E. H. Mamdani. Qualitative Dempster- Shafer theory. In *Proceedings of the IMACS III, International Workshop on Qualitative Reasoning and Decision Technologies*, pages 471–480, 1993.
18. J. Pearl and M. Goldszmidt. Qualitative probabilities for default reasoning, belief revision, and causal modeling. *Artificial intelligence journal*, 84(2):57–112, 1996.
19. L. J. Savage. *The foundations of statistics.* Wiley, New York, 1954.
20. C. E. Shannon and W. Weaver. *The mathematical theory of communication.* Urbana, University of Illinois Press, 1949.
21. M.P. Wellman. Qualitative probabilistic networks for planning under uncertainty. In J. F. Lemmer and L. N. Kanal, editors, *Uncertainty in Artificial Intelligence 2*, pages 197–217. Elsevier Science, 1988.

Qualitative Reasoning for Admission Control in ATM Networks[1]

Authors: J.L. Marzo, A. Bueno, R. Fabregat, T. Jove.
Institut d'Informàtica i Aplicacions. Universitat de Girona.
e-mail: {marzo, bueno, ramon, teo}@eia.udg.es

Abstract

Asynchronous Transfer Mode (ATM) provides transport functionality to the future Broadband Integrated Service Digital Network, ATM networks are connection oriented and Connection Admission Control (CAC) is a preventive control procedure responsible for determining whether a connection request is admitted or denied. There is a trade-off between the accuracy and the evaluation time on the CAC schemes. Most accurate methods spend excessive time on evaluations and consequently the request is rejected due to time-out. A possible approach is to evaluate some possible future situations (pre-evaluation). These results would be previously evaluated and saved on a table. Between consecutive connection requests, the evaluation system evaluates in advance the future response for each class of traffic.

The drawback occurs when two, or more, connection requests arrive close in time, so that the on-line pre-evaluation scheme has 'Unknown' on the response vector. The objective of this work is to cover the possible lack of information by using Qualitative Reasoning (QR). QR enables calculations with non-numerical descriptions and still producing coherent results.

1. Introduction

1.1 ATM Traffic Control Problems

The ATM transport network is based on fast packet switching using small fixed-size packets. ATM permits flexible bandwidth allocation, so that an important objective is to obtain the maximum statistical gain on a shared resource: the physical link.

CAC is a preventive control procedure responsible for determining whether a connection request is admitted or denied. The decision is based on the current network load, on the values of the traffic parameters, on the available network resources and on the required Quality of Service (QoS) of the existing connections and the new connection. QoS requirements are often formulated in terms of the constraints placed on the following network performance parameters: delay, delay variation and cell loss.

1.2 CAC Implementation Aspects

There is a trade-off between the accuracy and the evaluation time on the CAC schemes. Most accurate methods spend excessive time on evaluations and consequently, the request is rejected due to time-out. We propose the utilisation of the Enhanced Convolution Approach (ECA) in small buffer environments as evaluation method [FAB94b] and [MAR97]. The convolution approach also provides the Cell Loss Probability (CLP) for each class of traffic.

A possible approach is to evaluate possible future situations responses in advance (pre-evaluation). These results would be previously evaluated and saved on a table. A more dynamic approach is the on-line pre-evaluations scheme. In this case, a response vector is dynamically evaluated, containing values 'Yes', 'No' or 'Unknown' for each

[1] This work has been supported by CICYT (Spanish Education Ministry) under contract TIC95-0982-C02-

class of traffic. Between consecutive connection requests, the evaluation system evaluates in advance the future response for each class of traffic. More specifically, this means to evaluate all 'Unknown' values to obtain exact 'Yes' or 'No' values.

The required time to perform the CAC decision has to be reasonable. This requirement is particularly difficult to fulfil when the network is heavily loaded in terms of the number of multiplexed connections or when the multiplexed connections have different characteristics in terms of bit rates and burstiness. In this case the complexity of the calculations required to predict a cell loss value increases enormously.

2. Proposal: To Use QR on a CAC Algorithm

2.1 On-line Pre-evaluations

In order to offer a no-wait response to a new incoming demand, pre-evaluated calculations are investigated in [FAB94a] and [MAR95]. The inter-arrival time between calls ranges from a few seconds to minutes, during which time the evaluation sub-system may be idle. During this period the status of the system can be dynamically updated for each type of source, and restored to the actual status. After such evaluation the system knows if it can admit a new call for each type of source. A new pre-evaluated vector is used to store results:

$$R_0, R_1, \ldots, R_j, \ldots, R_{L-1}$$

Where $R_j \in$ {YES, NO}, and L denotes the number of traffic classes.

The evaluation sub-system always evaluates the CAC based on the new state of the system. Figure *1a* shows the ordinary scheme. The incoming demands ***a***, ***b***, and ***c*** obtain response with delay. The amount of this delay depends on the evaluation cost for this type of source. Figure *1b* shows the scenario with the proposed scheme. In this case, incoming demands ***a*** and ***b*** obtain an immediate response; the response to ***c*** is delayed because the evaluation sub-system is still evaluating the response vector *R* after the ***b*** connection. There is a certain probability of collision between a new connection demand and the last set-up connection process. This situation is analogous to the ordinary evaluation scheme in the worst case. The performance of this scheme depends on the process of connection arrivals and releases.

This proposal can be enhanced by adding a new possibility for the R_j state; the value UNKNOWN. This new response would be given in case of any time-out during the evaluation and means that the CAC has no response at this time.

Now $R_j \in$ {YES, NO, UNKNOWN}

That modification can save calculations when the arrival and departure demands are very dynamic. For example, when an incoming call or a disconnection arrives the pre-evaluation sub-system interrupts the background process and processes the new event.

In the pre-evaluation procedure, disconnection processes are considered to be all elements of the response vector *R* that are equal to NO. After the call termination, the responses of *R* equal to NO are transformed to UNKNOWN to be re-evaluated. Note that YES responses do not need to be updated because the system load has decreased. These responses will continue equal to YES. However, the connection set-up process, in a re-evaluation consist of all the elements of *R* that are equal to YES, so that those elements equal to YES are transformed in UNKNOWN. In this case the NO responses do not vary because the system load has increased.

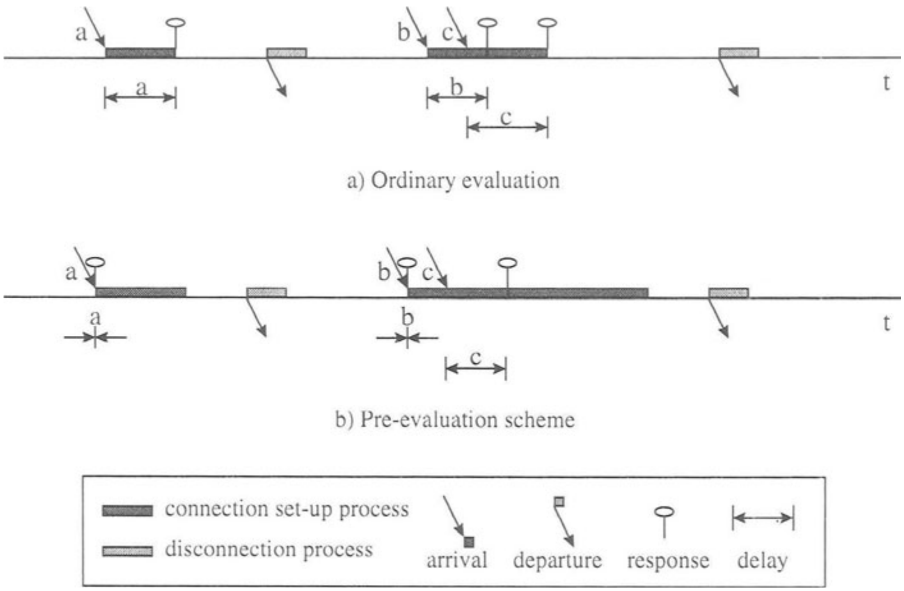


Fig. 1. Ordinary and Pre-evaluation CAC schemes

2.2 Qualitative Reasoning

Qualitative Reasoning (QR) is the expression of an old idea: to describe in a qualitative (not quantitative) way the environment. The classical approach for model construction requires an exhaustive knowledge of the modelled system, and a complete specification of the model variables. Many times the abstraction needed is excessive, resulting in a too distant model from reality, or too complex for the apparent simplicity of the situation. QR intends to apply a simpler approach for those systems without the need for a completely precise determination of their functioning. It could be seen as a trial to formalise the human common-sense. That is the reason why it is considered an useful tool for the Artificial Intelligence.

Human beings can solve many problems with their sheer experience and common-sense. For example, to determine the position of a mobile object, knowing its trajectory, the well known problem of velocity and acceleration can be easily solved. However, no-one uses this model to decide whether or not to cross the street. Almost everybody looks to one side, then to the other, and decides what to do, without making any equation. It is usual to see how engineers use their common-sense intuitions to resolve daily tasks.

One of the first times in which QR was used was in the 60's when economists discovered that they can obtain significant conclusions with just knowing the sign of some relevant parameters. It was the origin of the first QR mathematical model: the so-called Signs Algebra. In the 80's grew interest in QR to be applied in may fields like Physics, Automatic Control, Biology, ... The more complex a system was, the more attractive resulted the QR approach.

Initially, the main reason to use QR was the lack of numerical data. Other reasons can be given by the more suitable qualitative approximations for complex models (like those coming from AI, trying to approach machine and human reasoning), or the high computational cost of some of the decision making models based on large amounts of quantitative knowledge, which is the case of this paper.

2.3 A Wider Label Space

The first approach is inspired in the Signs Algebra: we consider three possible answers to a connection or disconnection request, *Y*es, *N*o and *U*nknown. When a connection request from a class marked as "Y" arrives to the system (table *1a*), the connection is accepted and this class becomes marked as "U". If the type of source is marked as "N", the connection is immediately rejected; this label only changes if a disconnection occurs.

CONNECTION	
⟶	
Y	U
N	N

Table 1a

DISCONNECTION	
⟶	
Y	Y
N	U

Table 1b

Transition tables

In case of a disconnection (table *1b*), if it comes from a class marked as "Y", it will remain with the same label. This is the formalisation of common-sense reasoning: there is space for a new connection at least of the same class as the disconnected one.

But in this case we could think that the new "Y" is stronger than the old one. The same at the connection scheme: the new "N" is more restrictive than the old one. For this reason we decided to consider different levels of "yes" and "no", getting a wider label space: *A*bsolutely *Y*es, *Y*es, *M*aybe *Y*es, *U*nknown, *M*aybe *N*o, *N*o and *A*bsolutely *N*o.

$$\text{Now } R_j \in \{\ \text{AY, Y, MY, U, MN, N, AN}\ \}$$

To extend the *U* state is not necessary because the only way to answer is a complete evaluation, since we don't have any qualitative information.

Transition Tables

Since we have a better label space, we can differentiate between a normal "yes" and a stronger "yes" (like the previous one with the previous label space at disconnection). Now the new space tables should be found. For example, as we consider identical two "AY" we must avoid transitions like AY→AY at *connections*. The reason is that if a type of traffic obtains this label, it will *always* have *that* label (i.e. if two thousand of connections from an "AY" labelled class are demanded, they are immediately accepted). For the same reason, we must avoid loops like AY→Y→AY in the same table. We can also consider that if a determined connection can be accepted, so we can with a less (bandwidth) demanding one. With these conditions, the transition tables become:

CONNECTION	
⟶	
AY	Y
Y	MY
MY	U
MN*	MN
N*	N
AN*	AN

Table 2a

DISCONNECTION	
⟶	
AY	AY
Y	Y / AY#
MY	MY / Y#
MN	U
N	MN
AN	N

Table 2b

Extended transition tables

Labels marked with (*) are impossible values in case of a just connected source, because the connection has been accepted. These transitions do not give new information to the system in order to modify the labels of the other classes. Therefore, their labels remain unchanged.

The (#) marks that left labels are for every class with *more* bandwidth demand than the disconnected one. The system cannot obtain new information in this case. The right labels are for the disconnected class and the *less* demanded bandwidth classes.

The AY→AY transition at *disconnection* (table *2b*) is not a problem. If the system is absolutely sure it can accept one of these connections, it will also be sure to accept more connections. Note that transitions from an "U" label are not included in the tables. This is because it cannot be solved with a pure qualitative reasoning.

The application of these tables is what we call the *QR procedure.*

2.4 Best Label Algorithm

Tables are not the only qualitative information that can be used. The last table argumentation can be rewritten this way: if a type of source is labelled as "Y", every less demanded bandwidth type of traffic should be labelled with at least a "Y". This is true both at connection and disconnection time: The system is sure, at a certain level, to handle a new connection from a specific class. Therefore, it must be certainly sure to handle a connection from a less demanded bandwidth type of source at least at the same level. To fix notation we will consider that the sources are sorted decreasingly by their demanded bandwidth.

If we note by $L_i(t)$ the label of every class i, at step t, the previous reasoning is formalised in the following manner:

1. Calculate $L_k(t+1)$ (with expensive calculation; see 2.4)
2. $L_i(t+1) = Max\ \{L_k(t+1), L_i(t)\}\ \forall i<k$ (i.e. for every source i with less bandwidth demand than source k)

Best Label Algorithm

Example:

Suppose there are two classes, type 0 and type 1, sorted as before (the first one with higher bandwidth demand than the second one). There are also cn_i connections from source i.

	Connections	**Label [1]**	**Label [2]**	**Label [3]**
Type of source 0	(cn_0+1, cn_1)	U —	→ Y	Y
Type of source 1	(cn_0, cn_1+1)	MY	MY —	→ Y

Table 3

The initial labels are at [1]. They come from qualitative operations. As the system does not have any qualitative answer to a future type 0 connection (because of the "U" label), it must calculate the type 0 label quantitatively using the CAC. The exact label ("Y") for source 0 is obtained at [2]. Now, as Source 1 has less demanded bandwidth, its label can be inferred from the Source 0 label by qualitative reasoning (using the Best Label Algorithm).

Thereby the system has at [3] an actualised vector (Y, Y), where each component i would be the i source label after a connection of the same type. Later qualitative labels are updated by successive applications of the transition tables.

The application of this algorithm is what we call the *QR-BLA procedure.*

2.5 Adding the CLPj as quantitative decision parameters

As was mentioned before, when a type of source label becomes an "U", it cannot be solved by a pure qualitative reasoning. It can also be observed that labels derived from other labels are qualitative approximations. To complete and complement the qualitative part of this scheme, we propose the utilisation of the Cell Loss Probability (CLP) for each type of source, as quantitative parameter. CLPs can be calculated from the Enhanced Convolution Approach (ECA), as presented in [MAR97]. This is considered as the single Quality of Service parameter. Other methods, proposed by literature, could also be used as quantitative decision parameter; for instance, the equivalent bandwidth [GUE91].

2.6 Partitioning the CLP Interval

The CAC system needs to evaluate the CLPs in case of "U" label (the exact label for a type of source must be evaluated). The interval of probabilities [0, 1] can be conveniently partitioned in prefixed intervals corresponding to each label. When an exact CLP is evaluated, the corresponding label is also updated into the response vector (*R*) for this class of traffic. Doing this, CLPs can obtain better labels than those qualitatively inferred (see ex. 1). Of course, this would only happen if the system does not have a higher priority process (a connection or a disconnection request).

The main difficulty when partitioning the interval of probabilities is to decide the criteria to do it. The CLP (in which we base the label assignation) rely upon the number of connections and the sources and channel characteristics. In short, too many variables to have a fixed partition. If it is made so, it can only expect to be too conservative or too risky.

In consequence, the working scheme is the following:

1. Each time that a type *i* **connection** is demanded, the label vector is consulted:

   1.1. If the *i* label is not "U", the connection is immediately accepted or rejected. The whole vector is actualised with the transition table (*QR procedure*).

   1.2. If the *i* label is equal to "U", the system needs to evaluate the CLP_i. Therefore, the exact label is determined using the partitioned interval and the whole vector is re-evaluated with the Best Label Algorithm described in 2.3. (*QR-BLA procedure*). Note that this is an exceptional situation, which is only possible when the interval of time between consecutive events (arrivals and departures) is too small.

2. Each time a type *i* **disconnection** is demanded, it is accepted and the label vector is updated with the transition table (*QR procedure*).

3. While the system is **not occupied,** it calculates quantitatively the exact label for each source using the CLPs obtained from the ECA. The U-labelled sources must be evaluated before the others. Each time an exact label is evaluated, the Best Label Algorithm is applied to re-evaluate the whole vector (*QR-BLA procedure*).

This background procedure (CLP calculation) is what we call *QC procedure* (for "quantitative calculation").

Same Example Using CLP$_j$

The previous example can be rewritten as follows:

	Connections	Label (CLP) [1]	Label (CLP) [2]	Label (CLP) [3]
Source Type 0	(cn_0+1, cn_1)	U (unknown)	→Y (known)	Y (known)
Source Type 1	(cn_0, cn_1+1)	MY (unknown)	MY (unknown)	→Y (unknown)

Table 4

The initial labels at [1] are the same. Their CLPs are supposed to be unknown because they have been obtained from qualitative operations. Now the system must calculate type 0 label quantitatively using the CAC. The resultant exact label "Y" is obtained, as shown in [2], when its CLP is exactly calculated, so its corresponding CLP is known. As type 1 has less demanded bandwidth, its label is inferred at [3] from the Source 0 label. As it is a qualitative reasoning, its CLP remains unknown. If the system does not receive a new connection demand for a while, this CLP will be calculated (and maybe improve the qualitative label).

Previous argumentations are summarised in the following scheme:

$$\left.\begin{array}{r}\text{LABEL j } \textit{(qualitatively estimated)}\\ +\ \text{CLP } \textit{(ECA algorithm)}\end{array}\right\} \Rightarrow \begin{array}{l}\text{LABEL j } \textit{(exact)}\\ \text{LABELS i}\\ \textit{.(qualitatively estimated)}\ \forall\ i\neq j\end{array}$$

2.7 Some Cases

Following there are some graphical examples of selected situations where the method performance can be observed. The horizontal axis represents *time* and CLP calculation time are marked over it.

The application of any of the three procedures: *QR procedure* (transition tables application), *QC procedure* (CLP calculation and label assignment) and *QR-BLA procedure* (Best Label Algorithm application) is respectively represented by **QR**, **QC** and **QR-BLA** tags.

The response vector *R* is represented between brackets. For example, **[U, MY, MY]** means an "U" label for type of source 0 (the one with more bandwidth demand) and a "MY" label for types 1 and 2. When a label is marked with an asterisk (like in **[MY, MY*, U]**), it means this label has been obtained by assignment after a CLP calculation and, consequently, the corresponding CLP is known.

Example 1: Making Use of Current Calculation

In this case a connection demand arrives when its corresponding CLP is being calculated. The system continues with it. When calculus is finished, the corresponding label is assigned and the connection is consequently accepted or rejected (in this case, at $t=t_2$, it is accepted).

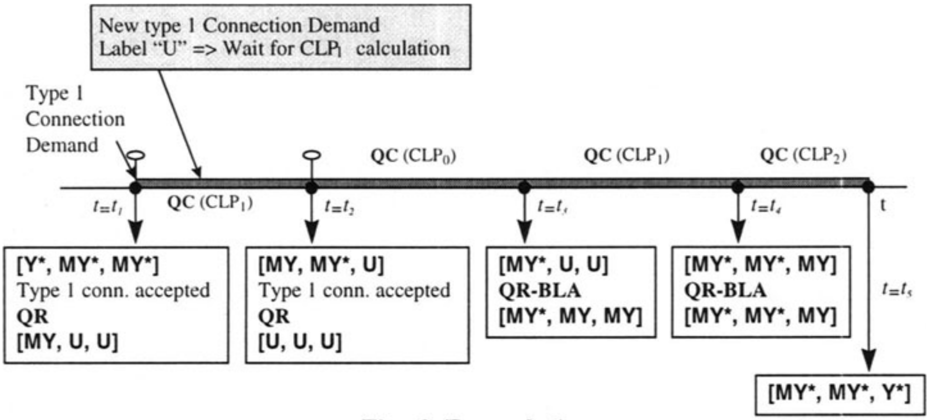


Fig. 2. Example 1

At $t=t_0$ the connection is immediately accepted and the system calculates CLP$_1$ as before. At $t=t_1$ the just calculated CLP allow to accept the new connection (The "MY" label is arbitrary to make shorter the example). The QR procedure turns every label an "U". Then the CLP$_0$ is calculated. At $t=t_2$ the new label and the QR-BLA give labels for every font. The system has calculated one CLP but has (qualitative) response for all sources. It calculates CLP$_1$. At $t=t_3$ the CLP calculation confirm the previously assigned label. The QR-BLA doesn't give new information. At $t=t_4$ the CLP calculation allows to assign a better label to source 2. The system has every CLP calculated.

Example 2: Making Use of Current Calculation (II)

In this case a connection demand arrives when a CLP from a higher demanded bandwidth is being calculated. This can affect to the incoming connection "U" label through the BLA algorithm. The system continues with calculus. When calculus is finished, the corresponding label is assigned, the QR-BLA procedure is applied and the connection is consequently accepted or rejected (in this case, at $t=t_2$, it is accepted). If the assigned label would have been a rejecting label, the QR-BLA would not have changed the incoming connection "U" label. Therefore its CLP should be calculated to resolve the request.

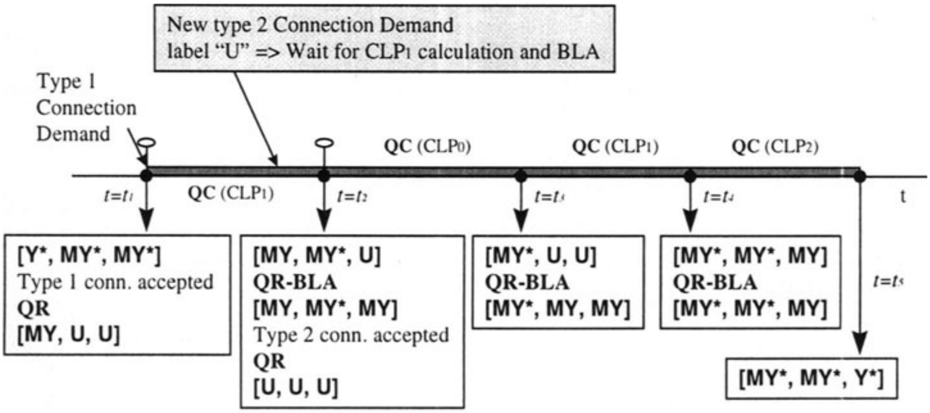


Fig. 3. Example 2

At $t=t_1$ the calculated CLP allows to QR-BLA assign a "MY" label. Now the system can accept the incoming connection without its corresponding CLP. After the QR procedure all the labels are "U". At $t=t_2$ the QR-BLA prepares responses for every source. At $t=t_3$ and $t=t_4$, successive CLP calculations improve the qualitative labels.

Example 3: CLP Calculation Aborted

In this example the connection request is from a higher bandwidth demand than the one whose CLP is being calculated. Then the calculus is aborted and the label vector is consulted. In this case a response is prepared and the connection is accepted. If the corresponding label would have been an "U", the necessary CLP is calculated. When it is done, the corresponding label is assigned and the connection is consequently accepted or rejected.

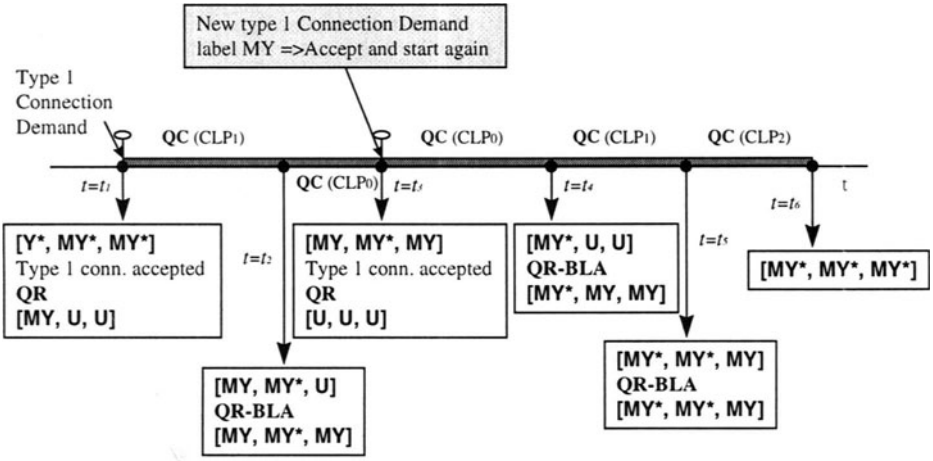


Fig. 4. Example 3

At $t=t_2$ the new connection is immediately accepted. The CLP_1 calculation is aborted and the process is restarted. At $t=t_3$ the QR-BLA prepares responses for every source after one CLP calculation. These responses are confirmed at $t=t_4$ and $t=t_5$ by the exact CLPs.

2.8 Example 4: disconnection

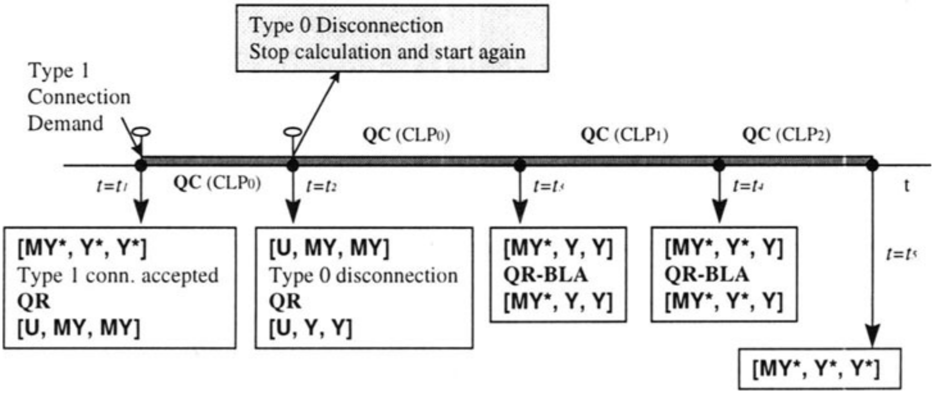


Fig. 5. Example 4

This example shows the usual evolution of a disconnection. When the disconnection occurs, CLP calculation is aborted, transition table *(1b)* is applied and calculus is restarted. This is the only effect that our scheme shows at disconnection. It is because every time a disconnection occurs, any calculation being made should be cancelled (and restarted if its label is an "U").

At $t=t_1$ the disconnection is immediately accepted. The labels are updates following the tables. Note that in spite of no having qualitative information about source 0, we get *better* information about the other sources. At $t=t_2$, after calculating one CLP, the system is again ready to answer to every demand. At $t \geq t_3$, as the system is unoccupied, the remaining CLPs are successively calculated. In this occasion, the QR-BLA does not give extra information.

2.9 Partitioning the CLP Interval: an Example

The complete CLP table is taken as base material for this example. It has been previously said that choosing a fixed (constant) partition of the CLP interval involves a compromise between a conservative and a risky point of view. In this case, to be risky means accept connections that wouldn't be accepted if only CLPs (w/o QR) were used. Then the Quality of Service (QoS) of the whole system is compromised: If a CLP is too high, it makes the system to loose too many data cells (remember that CLP means *Cell Loss Probability*), so every connection is affected by the wrongly accepted ones.

This is why the most conservative choice is developed in this section. The following values have been chosen to avoid wrong acceptance of connections.

$$\text{Label}_0 = \begin{cases} \text{AY} & \text{if} \quad CLP_0 \in [0 \;, 10^{-8}] \\ \text{Y} & \text{if} \quad CLP_0 \in]10^{-8}, 10^{-7}] \\ \text{MY} & \text{if} \quad CLP_0 \in]10^{-7}, 10^{-6}] \\ \text{MN} & \text{if} \quad CLP_0 \in]10^{-6}, 10^{-5}] \\ \text{N} & \text{if} \quad CLP_0 \in]10^{-5}, 10^{-4}] \\ \text{AN} & \text{if} \quad CLP_0 \in]10^{-4}, \; 1] \end{cases} \qquad \text{Label}_1 = \begin{cases} \text{AY} & \text{if} \quad CLP_1 \in [0 \;, 10^{-7}] \\ \text{Y} & \text{if} \quad CLP_1 \in]10^{-7}, 10^{-6}] \\ \text{MY} & \text{if} \quad CLP_1 \in]10^{-6}, 10^{-5}] \\ \text{MN} & \text{if} \quad CLP_1 \in]10^{-5}, 10^{-4}] \\ \text{N} & \text{if} \quad CLP_1 \in]10^{-4}, 10^{-3}] \\ \text{AN} & \text{if} \quad CLP_1 \in]10^{-3}, \; 1] \end{cases}$$

Table 6 : CLP_j interval partition

The conservative approach can be appreciated when more than one consecutive CLP becomes the same label. This can be appreciated by observing the CLP_1 when there is only connections from source 1 (the most conservative "side" of the CLP table):

Type 1 connections	CLP_1	label assigned
29	3,17E-07	Y
30	4,99E-07	Y
31	7,66E-07	Y
32	1,15E-06	MY
33	1,70E-06	MY
34	2,46E-06	MY
35	3,51E-06	MY
36	4,93E-06	MY
37	6,82E-06	MY
38	9,31E-06	MY
39	$>10^{-5}$	MN

The ideal situation would be if each label would appear just one time (with exception of the limit labels, AY and AN). Having some consecutive "MY"s means take advantage of the QR reasoning only in the 50% of the connection requests, because of the sequence MY→U→MY→U→...

In the case of having some consecutive "Y"s means capitalise on the QR only the 67% of the connections requests, because of the sequence Y → MY → U → Y → MY → U → Y → ...

The result is not so extreme if we look to the CLP_1 when we have *nine* connections from source 0:

Source 1 connections	CLP_1	label assigned
4	1,01E-08	AY
5	5,27E-08	AY
6	1,68E-07	Y
7	4,19E-07	Y
8	9,00E-07	Y
9	1,74E-06	MY
10	$>10^{-5}$	MN

Finally, when we arrive to the 12th connection from source 0 (with 13 connections we can accept no connection from source 1), the perfect label assignment, i.e. different labels for consecutive number of connections, is achieved:

Source 1 connections	CLP_1	label assigned
0	0	AY
1	4,99E-07	Y
2	7,66E-07	MY
3	$>10^{-5}$	MN

Any (fixed) refinement made to this partition would get a better label assignment with fewer connections from source 0. However, this would have the cost of wrongly accepted calls when having more source 0 connections.

3. Conclusions and Future Work

In this paper, the CAC with on-line pre-evaluations is extended with Qualitative Reasoning. We propose the application of QR to achieve fast response to an incoming demand in the case of a complex scenario that implies a long evaluation time.

The method is based on two mechanisms, a foreground process and a background process. The foreground process determines the response to an incoming demand based on the pre-evaluated vector. If the label is in the 'yes' set, the connection is accepted and the vector updated (based on QR rules). In the case of disconnection, the vector is also upgraded.

Between arrivals and releases, the CAC system is not idle and the incomplete values in the response vector are evaluated. Two processes are necessary: first, the CLP is completely calculated; secondly, the existing labels in the vector are updated corresponding to the evaluated CLP. Moreover, a label, in any type of traffic, may modify other response labels using QR.

The paper shows how QR can save evaluations when a fast response is needed. However, QR adds risk to the response. This is a work in progress, since the quantification of this risk in different scenarios is an objective for future work.

Another important subject to study is the development of a dynamical method to determine the optimal partition of the interval [0,1] of probabilities for the CLP-to-label conversion in the *QR-BLA procedure.* If this partition can be dynamically completed in terms of the connections number, and source and channel characteristics, the scheme-efficiency should be considerably improved.

The use of this approach in routing environments will also be an interesting future research project.

4. References

[AGE95] N. Agell, J.C. Aguado, M. Domènech, M. Sánchez and N. Piera, . "Mathematical Foundations of Qualitative Reasoning" in "Current trends in Qualitative Reasoning and Applications", pp. 12-24, International Center for Numerical Methods in Engineering, 1995.

[ESC95] M.T. Escrig, F. Toledo and A. P. del Pobil. "An Overview to Qualitative Spatial Reasoning" in "Current trends in Qualitative Reasoning and Applications", pp. 43-60, International Center for Numerical Methods in Engineering, 1995.

[FAB94a] Ramón Fabregat-Gesa, Josep Sole-Pareta, Jose Luis Marzo-Lázaro, Jordi Domingo-Pascual. "Adaptive Routing Based On Real-Time Calculations Using The Convolution Approach". EFOC & N'94.Twelfth Annual Conference on European Fibre Optic Communications and Networks. Heidelberg. June 21-24. 1994

[FAB94b] R. Fabregat-Gesa, J. Sole-Pareta, J.L. Marzo-Lázaro and J. Domingo-Pascual. "Bandwidth Allocation Based on Real Time Calculations Using the Convolution Approach", GLOBECOM'94, pp. 788-793.

[GUE91] Guerin R. and others. "Equivalent Capacity and its Application to Bandwidth Allocation in High-Speed Networks". IEEE Journal on Selected Areas in Communications, Vol 9, N° 7, September 1991.

[KUI94] B. Kuipers. "Qualitative Reasoning", The MIT Press, 1994

[MAR94] J. L. Marzo, J.Domingo, R. Fabregat, J. Sole. Eleventh UK Teletraffic Symposium. "Multicast Algorithms Evaluation Using An Adaptative Routing In ATM Networks", Performance Engineering in Telecomunications Networks. The Moller Centre. Cambridge. 23-25 March 1994.

[MAR95] J.L. Marzo, J. Domingo, R. Fabregat, J. Sole. "Dynamic routing based on a single parameter: Link Congestion Probability". High Performance Networking VI. IFIP sixth international conference on High Performance Networking. Palma, September 1995. pp. 307-318

[MAR97] J.L. Marzo. "Enhanced Convolution Approach for CAC in ATM Networks, an analytical study and implementation." Ph. D. Thesis. University of Girona. 1997

[MAR97b] J.L. Marzo, T. Jové, R. Fabregat, A. Bueno. "Connection Admission Control with on-line pre-evaluation: the extension with Qualitative Reasoning", 13 United Kingdom Workshop On Performance Engineering of Computer and Telecommunication Systems. Ilkley UK. 1997.

[R1022] "Updated Results of Traffic Simulation of the Policing Experiment". RACE1022 Technology for ATD. 1990.

Using Tolerance Calculus for Reasoning in Relative Order of Magnitude Models

Robert Dollinger and Ioan Alfred Letia

Department of Computer Science, Technical University of Cluj,
Gh.Baritiu 26, 3400-Cluj, Romania,
{Robert.Dollinger,Letia.Ioan.Alfred}@cs.utcluj.ro

Abstract. Order of magnitude (OM) reasoning is an approach that offers a midway abstraction level between numerical methods and qualitative formalisms. Relative OM models postulate a set of relations and inference rules on orders of magnitude. The main shortcoming of these models is the difficulty to validate the results they produce when applied to reasoning on real world problems. A widely accepted solution to avoid this deficiency is to extend the relative OM reasoning systems by a tolerance calculus. The way this extension is defined is a sensitive problem, affecting the accuracy of the results produced by the system. In this paper we present two ideas which could help to obtain more accurate results. First, we propose a more refined definition of the negligibility relation which is subject to avoid the artificial growth of tolerances. Second, we show that, in the case of many inference rules, one can derive tighter tolerance bounds when additional information is available.

1 Introduction

Order of magnitude (OM) formalisms have been introduced as alternatives to pure qualitative theories in order to address the problems deriving from the poor expressivity of these theories, leading to what is known as the overabstraction phenomenon [5]. In OM theories the level of abstraction is reduced by integrating information concerning the relative magnitude of the real values, thus providing an intermediate abstraction level between numerical methods and qualitative formalisms. OM models can be classified in two categories: **absolute OM models** (Dubois and Prade [3], Missier [7] and Missier, Pierra and Travé-Massuyès [8]), that are based on the definition of a set of primitive objects from which their manipulation rules are derived and, **relative OM models** that postulate a set of relations and a set of inference rules on orders of magnitude, without making any assumption on the nature of the objects consistent with these rules. The most important relative OM models are: FOG (Formal Order of maGnitude reasoning system) proposed by Raiman [9], O(M) introduced by Mavrovouniotis and Stephanoupoulos [6] and, finally, ROM(K) due to Dague [1].

One major problem that affects the relative OM models is the difficulty to control the inference process, in order to obtain valid results in the real world. The problem can be solved by extending these models with a tolerance calculus that provides a measure of accuracy for the inferred results. Such extensions were proposed first by Dubois and Prade for FOG [4] and more recently by Dague for ROM[K] [2]. There are slight differences in the way these extensions are defined in the two papers. These definitions are a very sensitive point with effects on the properties of the extended systems and on the quality of the results they produce. In this paper we propose some modifications in the definition of the negligibility relation subject to avoid the artificial introduction of errors, thus producing more accurate results. A second issue, we deal with, concerns the possibility to obtain more accurate results by making use of additional information. In section 2 we briefly review a tolerance calculus based OM reasoning model that was proposed by Dague [2]. In section 3 we propose some improvements in order to eliminate certain problems encountered in previous extensions of relative OM models. Section 4 is based on the idea of using additional information in order to improve tolerances. It shows how certain rules can be specialized in more precise subrules when additional sign or order of magnitude information is available. Section 5 illustrates the considerations from section 4 by using the heat exchanger example.

2 Extended Relative OM Models

The relative OM models are pure symbolic reasoning systems affected by two main disadvantages: difficulty to incorporate quantitative information and lack of control in the inference process.

Some rules in ROM(K) (the same goes for FOG to) are subject to produce invalid results when applied repeatedly in the same inference chain. Hence, it appears as a necessity to equip these systems with a semantics capable of providing a measure for the degree of satisfaction (tolerance) of any given relation between orders of magnitude. Also, for each new relation inferred in the reasoning process, we should be able to derive its tolerance in terms of the tolerances available for the initial relations. An extension satisfying the above requirements was first proposed by Dubois and Prade for Raiman's FOG system [4]. Their idea was reiterated by Dague who provided a similar extension, with slight differences, for the ROM(K) model [2]. The extension given by Dague is based on some parameterized definitions of the four relation operators. He simply postulates the following definitions:

Closeness: $x \text{ Vo}(\varepsilon)\, y \Leftrightarrow 1-\varepsilon \leq x/y \leq 1/(1-\varepsilon)$, where $0 < \varepsilon < 1$.
Negligibility: $x \text{ Ne}(\varepsilon)\, y \Leftrightarrow -\varepsilon \leq x/y \leq \varepsilon$, where $0 < \varepsilon < 1$.

Comparability: $x \text{ Co}(\varepsilon_1)\ y \Leftrightarrow x \text{ Vo}(\varepsilon_2)\ y$, where $1/2 < \varepsilon_1 < 1$ and $\varepsilon_1 < \varepsilon_2 < 1-\varepsilon_1$.

Distant: $x \text{ Di}(\varepsilon)\ y \Leftrightarrow x/y \leq \varepsilon$ or $x/y \geq 1/\varepsilon$ or $y = 0$, where $0 < \varepsilon < 1$.

Notice that the smaller a tolerance parameter is, the more accurate the associated order of magnitude relation.

Given the parameterized definitions of the four relation operators, one can derive a tolerance formula for each of the axioms A1-A15 in ROM(K). These formulas allow us to express the tolerance associated to each derived relation in terms of the tolerances given for the initial relations. Axioms A1-A15 with the associated tolerance formulas constitute the basis for an extended relative OM system with the following inference rules:

A1:	$x \text{ Vo}(\varepsilon)\ x$,	$\varepsilon=0$,	
A2:	$x \text{ Vo}(\varepsilon_1)\ y \rightarrow y \text{ Vo}(\varepsilon)\ x$,	$\varepsilon=\varepsilon_1$,	
A3:	$x \text{ Vo}(\varepsilon_1)\ y,\ y \text{ Vo}(\varepsilon_2)\ z \rightarrow x \text{ Vo}(\varepsilon)\ z$,	$\varepsilon=\varepsilon_1+\varepsilon_2-\varepsilon_1\cdot\varepsilon_2$,	
A4:	$x \text{ Co}(\varepsilon_1)\ y \rightarrow y \text{ Co}(\varepsilon)\ x$,	$\varepsilon=\varepsilon_1$,	
A5:	$x \text{ Co}(\varepsilon_1)\ y,\ y \text{ Co}(\varepsilon_2)\ z \rightarrow x \text{ Co}(\varepsilon)\ z$,	$\varepsilon=\varepsilon_1+\varepsilon_2-\varepsilon_1\cdot\varepsilon_2$,	
A6:	$x \text{ Vo}(\varepsilon_1)\ y \rightarrow x \text{ Co}(\varepsilon)\ y$,	$\varepsilon=\varepsilon_1$,	
A7:	$x \text{ Vo}(\varepsilon_1)\ y \rightarrow t\cdot x \text{ Vo}(\varepsilon)\ t\cdot y$,	$\varepsilon=\varepsilon_1$,	
A8:	$x \text{ Co}(\varepsilon_1)\ y \rightarrow t\cdot x \text{ Co}(\varepsilon)\ t\cdot y$,	$\varepsilon=\varepsilon_1$,	
A9:	$x \text{ Co}(\varepsilon_1)\ 1 \rightarrow [x] = +$,	____,	
A10:	$x \text{ Ne}(\varepsilon_1)\ y \rightarrow y \text{ Vo}(\varepsilon)\ (x+y)$,	$\varepsilon=\varepsilon_1$,	
	$y \text{ Vo}(\varepsilon_1)\ (x+y) \rightarrow x \text{ Ne}(\varepsilon)\ y$,	$\varepsilon=\varepsilon_1/(1-\varepsilon_1)$,	$\varepsilon_1<1/2$
A11:	$x \text{ Ne}(\varepsilon_1)\ y,\ y \text{ Co}(\varepsilon_2)\ z \rightarrow x \text{ Ne}(\varepsilon)\ z$,	$\varepsilon=\varepsilon_1/(1-\varepsilon_2)$,	$\varepsilon_1<1-\varepsilon_2$
A12:	$x \text{ Vo}(\varepsilon_1)\ y,\ [z] = [x] \rightarrow (x+z) \text{ Vo}(\varepsilon)\ (y+z)$,	$\varepsilon=\varepsilon_1$,	
A13:	$x \text{ Co}(\varepsilon_1)\ y,\ [z] = [x] \rightarrow (x+z) \text{ Co}(\varepsilon)\ (y+z)$,	$\varepsilon=\varepsilon_1$,	$\varepsilon_1>1/2$
A14:	$x \text{ Co}(\varepsilon)\ (x+x)$,	$\varepsilon=1/2$,	
A15:	$x \text{ Di}(\varepsilon_1)\ y \leftrightarrow (x-y) \text{ Co}(\varepsilon)\ x$ or $(y-x) \text{ Co}(\varepsilon)\ y$,	$\varepsilon=\varepsilon_1$,	

where $0 < \varepsilon, \varepsilon_1, \varepsilon_2 < 1$ and $[x]$ is the sign of x.

Similar extensions can be derived for the 45 properties presented by Dague in [1].

The extended system can be successfully applied to real problems as one can compute the corresponding tolerance for each relation derived in the reasoning process. A given result can be accepted or rejected according to the value of its associated tolerance. Moreover, the tolerance parameters can be used to control the inference process itself: one can prune a deduction chain whenever, at a certain step, the associated tolerance exceeds a given threshold, thus limiting the effects of the combinatorial explosion.

Analyzing the set of parameterized axioms one can observe that axiom A10 appears as a pair of implications with different associated tolerance formulas. Thus, as it can be seen, when transforming the Ne operator into Vo the tolerance is not changed, but the reversed transformation deteriorates the associated tolerance. This asymmetry appears somehow unnatural since in ROM(K) axiom A10 was aimed as a formal definition for the Ne operator. It also artificially introduces errors in the inference chains if, for some reason, one needs to convert the Ne operator into Vo and then back again into Ne. This could lead to unjustified deterioration of the tolerances, maybe causing some results to be rejected when, in fact, they should be considered as valid.

3 Modified Definition of the Ne Relation Operator

The cause of the above mentioned problems lies in the way the parameterized version of the Ne operator is defined. There are two different approaches for defining Ne. The first one, due to Dubois and Prade [4], is: x Ne(ε) y $\Leftrightarrow$ (y+x) Vo(ε) y and (y-x) Vo(ε) y from where they deduce: x Ne(ε) y $\Leftrightarrow$ $-\varepsilon \leq x/y \leq \varepsilon$. This definition is accurate only if $\varepsilon=0$ (i.e. at limit, where we enter in the framework of non-standard analysis), since the relations: (y+x) Vo(ε) y and (y-x) Vo(ε) y will never have the same best possible tolerance ε, unless $\varepsilon=0$. Indeed, if one accepts the first relation: (y+x) Vo y with tolerance ε, then for the second relation: (y-x) Vo y one shall have a tolerance ε_1 which depends on ε by the following relation $\varepsilon_1=\varepsilon/(1-\varepsilon)$, and it is easy to notice that $\varepsilon_1>\varepsilon$, if $\varepsilon>0$. The above definition does not exploit the fact that the relation (y+x) Vo y is tighter than (y-x) Vo y (assuming that [x]=[y]).

The second approach, adopted by Dague [2], is to simply postulate the definition of Ne in the form it was deduced by Dubois and Prade. But, as it was pointed out in the previous section, this is in contradiction with axiom A10 from ROM(K) which cannot be accepted any more as a formal definition for the Ne operator.

In order to eliminate all these problems we propose a modified form of axiom A10, let this be A10', and use it as a formal definition for the Ne operator both in unparameterized and parameterized form. Axiom A10' will remain an equivalence even in the extended relative OM model and tolerances will be preserved when transforming the Ne operator into Vo and viceversa. Thus, we introduce the following:

Definition: For any orders of magnitude x and y we have the equivalence:
A10': x Ne y $\leftrightarrow$ (y-x) Vo (y+x),
and we can prove the following:

Theorem: If the parameterized version of relation Vo is defined by

$$x \text{ Vo}(\varepsilon)\ y \Leftrightarrow 1-\varepsilon \leq x/y \leq 1/(1-\varepsilon), \quad \text{where } 0<\varepsilon<1,$$

and the relation operator Ne is defined by equivalence A10', then:

a) The parameterized version of Ne is:

$$x \text{ Ne}(\varepsilon)\ y \Leftrightarrow -\varepsilon/2/(1-\varepsilon/2) \leq x/y \leq \varepsilon/2/(1-\varepsilon/2),$$

b) The parameterized version of axiom A10' is:

A10': x Ne(ε) y $\leftrightarrow$ (y-x) Vo(ε) (y+x), where $0<\varepsilon<1$.

Proof: (omitted)

One can expect that the modified system will produce more accurate results each time transformations of Vo relations into Ne relations are involved. In the case of the reversed transformations (from Ne to Vo) the results should be at least as accurate as for the extended relative OM model of Dague. Nevertheless, it may be objected that a relation of the form y Vo (x+y) is more convenient to use in the reasoning process than (y-x) Vo (y+x). This is an arguable point and the answer depends very much on the context at hand. The worst case is when one has to transform a Ne relation into Vo and the context of the reasoning requires this last relation in the simpler form (y Vo (x+y)). This situation can be handled easily by adding a new rule to the system and by introducing an additional inference step in the reasoning chain. The new rule is of the form:

R: $(y\text{-}x)\ Vo(\varepsilon_1)\ (y+x) \rightarrow y\ Vo(\varepsilon)\ (y+x)$,

and can be inferred from axioms A12 and A7 and it does not appear in the set of properties introduced by Dague [2]. Its associated tolerance formula can be calculated directly and is $\varepsilon=\varepsilon_1/(2-\varepsilon_1)$. One can immediately notice that rule **R** has the property of enhancing the tolerance parameter, since $\varepsilon<\varepsilon_1$, for $0<\varepsilon_1<1$.

Further, it can be shown that transformation of a Ne relation into Vo by using axiom A10 or, alternatively, by using axiom A10' and rule R gives the same results (tolerances) if we take into account the differences in the semantics of the parameterized Ne relation for the two cases. Thus, even in the worst case, using axiom A10' results in tolerances that are at least as accurate as those produced by using axiom A10. These considerations show that by using axiom A10' instead of A10 and the modified definition of the Ne relation the results will, on average, be more accurate in any context.

4 Specialization of Rules

This section starts from the observation that one has the possibility to improve the inference system of a tolerance calculus based relative OM model by making use of additional information, when available, to deduce tighter bounds for the tolerance parameters. Incidentally, Dague [2] deduced tighter tolerances for some rules by assuming certain additional sign relations between orders of magnitude. Here, we go further with this idea by assuming not only sign information, but, especially, additional information about the relative magnitude of the quantities involved. When such information is available, some of the rules can be split in more specialized versions, producing subrules which may have much better estimates of the corresponding tolerances.

The inference system presented by Dague [1], consisting of axioms A1-A15 and properties P0-P44, contains many rules subject to be split in several specialized subrules with tighter tolerance formulas. One of these is axiom A12 for which, in the general case, $\varepsilon=\varepsilon_1$ is the best possible estimate for the tolerance of the resulting relation. However, if some additional information about the relationship between x and z is available, which is often the case, we can deduce different estimates for the tolerance of the inferred relation. These estimates depend upon the relation between x and z. We have 3 cases:

a) x $Ne(\varepsilon_2)$ z, which means that: $-\varepsilon_2 \leq x/z \leq \varepsilon_2$, but since [z]=[x], we have: $z/x \geq 1/\varepsilon_2$. Also, from the hypothesis: x $Vo(\varepsilon_1)$ y, we get: $1-\varepsilon_1 \leq x/y \leq 1/(1-\varepsilon_1)$, from where it can be observed that: [x]=[y]. Now, we can assume, without loss of generality, that: $x \geq y$, which means that: $z/y \geq z/x \geq 1/\varepsilon_2$, and more with [z]=[x]=[y] we have: $(x+z)/(y+z) \geq 1$. Thus in order to calculate the tolerance for the relation (x+z) $Vo(\varepsilon)$ (y+z) it is sufficient to find the upper estimate of the expression: $(x+z)/(y+z)$. But, $(x+z)/(y+z)=1+(x/y-1)/(1+z/y)$, thus by replacing with the upper estimate of x/y and the lower estimate of z/y, we get: $(x+z)/(y+z) \leq 1/(1-\varepsilon_1 \cdot \varepsilon_2/(1-\varepsilon_1+\varepsilon_2))$. This inequality shows that the relation (x+z) $Vo(\varepsilon)$ (y+z) is satisfied with tolerance:

$$\varepsilon=\varepsilon_1 \cdot \varepsilon_2/(1-\varepsilon_1+\varepsilon_2).$$

We notice that this is a second order expression in terms of the tolerances ε_1 and ε_2, and has much smaller values than ε_1 when ε_2 is close to 0.

b) x $Vo(\varepsilon_2)$ z, which means that: $1-\varepsilon_2 \leq x/z \leq 1/(1-\varepsilon_2)$. By a similar argument to that of point a) we get: $\varepsilon=\varepsilon_1/(1+(1-\varepsilon_1)\cdot(1+\varepsilon_2))$. Notice that if ε_1 and ε_2 are close to 0, $\varepsilon \approx \varepsilon_1/2$.

c) z $Ne(\varepsilon_2)$ x, in this case the resulting tolerance is: $\varepsilon=\varepsilon_1$, the same as in the general case.

Another axiom for which similar results can be obtained is A13.

The use of sign information leads to two other less spectacular cases where enhancements can be brought; these are axioms A3 and A5. If [x-y]=[z-y] then the resulting tolerance is: $\varepsilon=\max(\varepsilon_1,\varepsilon_2)$, instead of $\varepsilon=\varepsilon_1+\varepsilon_2-\varepsilon_1 \cdot \varepsilon_2$, for both rules.

Among properties P0-P44 there are several rules that have special cases when they produce more accurate results than shown by the general tolerance formula. The first of these rules is:

P6: x $Ne(\varepsilon_1)$ z, y $Ne(\varepsilon_2)$ z → (x+y) $Ne(\varepsilon)$ z, $\varepsilon=\varepsilon_1+\varepsilon_2$.

As shown by Dague [2], if [x]=-[y] then the resulting tolerance is: $\varepsilon=\max(\varepsilon_1,\varepsilon_2)$, which is obviously better than the general one.

We list bellow some other rules of Dague's system which, like axioms A12 and A13, produce more accurate results in the presence of additional information:

P16: x $Ne(\varepsilon_1)$ y, [z]=[y] → x $Ne(\varepsilon)$ (y+z), $\varepsilon=\varepsilon_1$,
we have: $\varepsilon=\varepsilon_1 \cdot \varepsilon_2/(1+\varepsilon_2)$, if y $Ne(\varepsilon_2)$ z,
and $\varepsilon=\varepsilon_1/(2-\varepsilon_2)$, if y $Vo(\varepsilon_2)$ z.

P20: (x+z) $Ne(\varepsilon_1)$ (y+z), [z]=[x] → x $Ne(\varepsilon)$ y, $\varepsilon=\varepsilon_1$,
we have: $\varepsilon=\varepsilon_1 \cdot \varepsilon_2/(1-\varepsilon_1)$, if x $Ne(\varepsilon_2)$ z,
and $\varepsilon=\varepsilon_1/(2-\varepsilon_2-\varepsilon_1/(1-\varepsilon_1))$, if x $Vo(\varepsilon_2)$ z.

P23: (x+z) $Ne(\varepsilon_1)$ y, [x]=[z] → x $Ne(\varepsilon)$ y, $\varepsilon=\varepsilon_1$,
we have: $\varepsilon=\varepsilon_1 \cdot \varepsilon_2/(1+\varepsilon_2)$, if x $Ne(\varepsilon_2)$ z,
and $\varepsilon=\varepsilon_1/(2-\varepsilon_2)$, if x $Vo(\varepsilon_2)$ z.

P42: (x+z) Di(ε_1) (y+z), [z]=[x] → x Di(ε) y, $\varepsilon=\varepsilon_1$,
we have: $\varepsilon=\varepsilon_1\cdot\varepsilon_2/(1-\varepsilon_1)$, if [y]=[x] and x Ne($\varepsilon_2$) z,
and $\varepsilon=\varepsilon_1/((2-\varepsilon_1-\varepsilon_2)\cdot(1-\varepsilon_2))$, if [y]=[x] and x Vo($\varepsilon_2$) z.
If [y]=-[x] then rule P42 is trivially satisfied with tolerance 0.

Use of sign information can bring more or less significant improvements in the case of some other rules like: P9, P10, P31 and P32.

Finally, we point out a less obvious aspect which is an interesting consequence of the above considerations. Any reasoning made with a tolerance calculus based relative OM system, like the one analyzed here, starts with a few order of magnitude relations having some given tolerances. During the reasoning process the tolerances are usually degraded, which means that with each step the inferred results become less accurate. Even if undesirable, this is a natural tendency, since all the rules of Dague's inference system, except property P11, produce relations which have tolerances greater than or at best equal to the tolerances associated to the premises. On the other hand, most of the specialized rules we listed above have the property of enhancing the resulting tolerances. Thus, by using these rules when possible, the degradation tendency of the tolerances will be diminished. This has the advantage of increasing the chances to obtain an acceptable solution, when the admissible criterion is to have a sufficiently small value of the associated tolerance parameter.

5 The Heat Exchanger Example

Let us illustrate the above considerations by the already classical example of the counter curent heat exchanger used by Mavrovouniotis and Stephanopoulos [6] and by Dague [2]. The main parameters of this physical device are: FH,FC - the molar flowrates of the hot and cold streams, KH,KC - the molar heats of the hot and cold streams, and the following temperature differences: DTH - the temperature drop of the hot stream, DTC - the temperature rise of the cold stream, DT1 - the driving force at the left end of the device, DT2 - the driving force at the right end of the device. Observe that, as defined, all the above parameters are positive quantities.

The definition of the four temperature differences gives the following equation:
(1) DTH-DT1-DTC+DT2=0.

We also assume the following, reasonable, qualitative relations:
(2) DTH Vo(ε_1) DTC,
(3) FH Vo(ε_2) FC (similar flowrates for the hot and cold streams),
(4) KH Vo(ε_3) KC (same liquid composition for the hot and cold streams),
(5) DT1 Ne(ε_4) DTH.
Our purpose is to estimate the energy balance of the heat exchanger. From (2),(3) and (4) we get:
(6) DTH·FH·KH Vo(ε) DTC·FC·KC (A7,A7,A3,A7,A7,A3),
where: $\varepsilon=\varepsilon_1+\varepsilon_2+\varepsilon_3-\varepsilon_1\cdot\varepsilon_2-\varepsilon_1\cdot\varepsilon_3-\varepsilon_2\cdot\varepsilon_3+\varepsilon_1\cdot\varepsilon_2\cdot\varepsilon_3$.

Now, let's suppose the input temperature of the hot stream rises with an amount ΔT. This means that relation (2) will be replaced by
(2') DTH+ΔT Vo(ε'_1) DTC+ΔT, (see equation (1) and relation (5)).

As shown in the previous section (axiom A12) the value of the tolerance parameter ε'_1 depends on the order of magnitude relation between DTH and ΔT. If ΔT Ne(ε_5) DTH then $\varepsilon'_1=\varepsilon_1$ and with $\varepsilon_1=0.2$, $\varepsilon_2=\varepsilon_3=0.1$, we get the following tolerance for relation (6): $\varepsilon\approx0.36$. If ΔT Vo(ε_5) DTH then $\varepsilon'_1=\varepsilon_1/(1+(1-\varepsilon_1)\cdot(1+\varepsilon_2))$, and if we, additionaly, consider $\varepsilon_5=0$, then we get: $\varepsilon\approx0.28$. Finally, if DTH Ne(ε_5) ΔT then $\varepsilon'_1=\varepsilon_1\cdot\varepsilon_5/(1-\varepsilon_1+\varepsilon_5)$ and if, for example, we consider $\varepsilon_5=\varepsilon_1$, then we get: $\varepsilon\approx0.2$.

The tolerance, ε, of the energy balance relation (6) can be considered as a measure of the heat exchanger efficiency. When ε is close to zero the efficiency is high, if ε increases the efficiency degrades. The 3 values obtained above for ε demonstrate the fact that the efficiency of the heat exchanger gets better when the input temperature of the hot stream is rised.

6 Conclusions

We have revealed some weaknesses in the definition of the parameterized Ne relation as it was introduced in the extended relative OM models proposed by Dubois and Prade [4] and by Dague [1,2]. These weaknesses combined with the asymmetric form of axiom A10 may cause a spurious deterioration of the results produced in a reasoning process. In order to avoid such problems we proposed a new axiom A10' which is symmetric and can be taken as the basis for the definition of the parameterized Ne relation. This provides a sound and coherent formal definition for the Ne relation, in terms of the Vo relation, which extends naturally to the parameterized version of the Ne relation as well. We also brought some arguments that the results produced by a reasoning system using the proposed modifications will not be less accurate in any context and are expected to be more accurate in many cases. In section 4 it was shown how additional sign or order of magnitude information can lead to more accurate results. By making use of such information many inference rules can be split in several specialized versions, most of them having the property of enhancing the tolerance parameter. This reduces the natural, but undesirable tendency of tolerance degradation in any reasoning process.

References

[1] Dague,P. - *Symbolic Reasoning with Relative Orders of Magnitude*, Proceedings of the Thirteenth IJCAI, Chamberry, (1993), p.1509-1514.

[2] Dague,P. - *Numeric Reasoning with Relative Orders of Magnitude*, Proceedings of AAAI Conference, Washington, (1993), p.541-547.

[3] Dubois,D., Prade,H. - *Fuzzy arithmetic in qualitative reasoning*,

Modelling and Control or Systems in Engineering, Quantum Mechanics, Economics and Biosciences, Proceedings of Bellman Continuum Workshop 1988, Sophia Antipolis, France, (1989), p.457-467.

[4] Dubois,D., Prade,H. - *Semantic considerations on order of magnitude reasoning*, Decision Support Systems and Qualitative Reasoning, Elsevier Sci.Publ.B.V. (North-Holland), IMACS, (1991), p.223-228.

[5] Kuipers,B.J. - *Qualitative simulation*, Artificial Intelligence, 29, (1986), p.289-338.

[6] Mavrovouniotis,M.L., Stephanopoulos,G. - *Reasoning with orders of magnitude and approximate relations*, Proceedings of the 6th National Conference on Artificial Intelligence, (AAAI 1987), Seattle, WA, (1987), p.626-630.

[7] Missier, A. - *Structures Mathèmatiques pour le Calcul Qualitatif, Contribution à la Simulation Qualitative*, Ph.D. thesis LAAS, Dec. 1991, Toulouse, France.

[8] Missier,A., Piera,N., Travé-Massuyès,L. - *Mathematical structures for qualitative calculus*, personal communication, (1992).

[9] Raiman,O. - *Order of magnitude reasoning*, Proceedings of the 5th National Conference on Artificial Intelligence (AAAI 1986), Philadelphia, PA, (1986), p.100-104.

Complexity and Cognitive Computing

Lourdes Mattos Brasil[1,3], Fernando Mendes de Azevedo[1], Jorge Muniz Barreto[2], and Monique Noirhomme- Fraiture[3]

[1] Dept. of Electrical Engineering, Federal University of Santa Catarina
University Campus, Florianópolis, Brazil, 88040-900
{lourdes†, azevedo}@gpeb.ufsc.br
[2] Dept. of Informatics and Statistics, Federal University of Santa Catarina
University Campus, Florianópolis, Brazil, 88040-900
barreto@inf.ufsc.br
[3] Institut d'Informatique, FUNDP,
Rue Grandgagnage, 21 B-5000 Namur (Belgium)
{lma, mno}@info.fundp.ac.be

Abstract. This paper proposes a hybrid expert system to minimize some of the complexity problems present in the artificial intelligence field such as the so-called bottleneck of expert systems, e.g., the knowledge elicitation process; the model choice for the knowledge representation to code human reasoning; the number of neurons in the hidden layer and the topology used in the connectionist approach, ; the difficulty to obtain the explanation on how the network arrived to a conclusion. To overcome these difficulties the cognitive computing was integrated to the developed system.

1 Introduction

In the last decade, a new area has emerged without the Artifial Intelligence (AI) field, the so-called cognitive computing. Cognitive computing is a collection of emerging technologies inspired by the biological processing of information in the nervous system, the human reasoning, the decision making, and the natural selection. Cognitive computing draws on the new discoveries being made in neuroscience, cognitive science, and biology, as well as the rapid advances underway in computing technology. Cognitive computing takes also advantaged of the present knowledge in statistics, classical control theory, signal and image processing and AI. As traditional computing technologies are quantitative in nature and emphasise precision and sequential order, cognitive computing tries to exploit the tolerance for imprecision, uncertainty and partial truth found in biological systems to achieve tractability, robustness and low cost solutions to engineering problems [1].

Cognitive computing is based on several related and emerging areas, including Artificial Neural Networks (ANN), fuzzy logic, evolutionary computation,

† Currently at the Institut d'Informatique, FUNDP, Namur, Belgium: the work sponsored by CAPES, Brazil.

probabilistic reasoning and chaos theory [1]. It is important to note that cognitive computing is a partnership of technologies where each partner contributes with a different methodology to address problems in its domain. ANN use the interactions of biological neurons as a model for pattern recognition, decision, modelling, and forecasting. Fuzzy logic uses information in a manner similar to the human decision process and it is useful in control and decision making applications. Evolutionary computation is based on natural selection and evolution and is useful in optimisation [2].

Using the cognitive computing, it was developed in this work a Hybrid Expert System (HES). This system consists of a Neural Network Based System Expert (NNES), a Rule Based Expert System (RBES) and, an Explanatory Expert Systems (EES).

2 Artificial Neural Network, Fuzzy Logical, and Evolutionary Computation

2.1 Artificial Neural Network (ANN)

An ANN (also known as connectionist architectures, connectionist paradigms, parallel distributed processing, and neuromorphic systems) is an information processing paradigm inspired by the way that the parallel structure of the mammalian brain processes information. ANN are collections of mathematical models that emulate some of the observed properties of biological nervous systems and draw on the analogies of adaptive biological learning. The key element of the ANN paradigm is the novel structure to process information. It is composed of a large number of highly interconnected processing elements that are analogous to neurons and which are tied together through weighted connections like the synapses [3].

Learning in biological systems involves adjustments to the synaptic connections that exist between the neurons. This is true of ANN as well. Learning typically occurs by example through training, or exposure to a truthed set of input/output data where the training algorithm iteratively adjusts the connection weights (synapses). These connection weights store the knowledge necessary to solve specific problems.

Although ANN have been around since the late 1950's, it was not until the mid-1980's that algorithms became sophisticated enough to allow their useful applications. Today ANN are applied to an increasing number of complex real-world problems. They are very useful in pattern recognition engines and robust classifiers, having the ability to generalize in making decisions about imprecise input data. They offer ideal solutions to a variety of classification problems such as in speech, character and signal recognition, as well as in functional prediction and system modelling where the physical processes are very complex or not well understood. ANN may also be applied to control problems, where the input variables are measurements used to drive an output actuator, and the network learns the control function.

The advantage of ANN lies in their resilience against distortions in the input data and their capability of learning. They are often good at solving problems that are too complex for conventional technologies (e.g., problems that do not have an algorithmic solution or for which an algorithmic solution is too complex to be found) and are often well suited to problems that people are good at solving, but for which traditional methods are not [3].

There are several types of ANN. One of the most popular is the multilayer perceptron which is generally trained with the backpropagation error algorithm [4]. In fact this algorithm inspired in the classical backpropagation one that is used in this work. This algorithm is described in the next sections.

2.2 Fuzzy Logical

Many decision-making and problem-solving tasks are too complex to be understood quantitatively, however, people succeed by using knowledge that is imprecise rather than precise. Fuzzy set theory, originally introduced by Lotfi Zadeh in the 1960's, resembles human reasoning in its use of approximate information and uncertainty to generate decisions. It was specifically designed to mathematically represent uncertainty and vagueness and provide formalized tools for dealing with the imprecision intrinsic to many problems. By contrast, traditional computing demands precision down to each bit. Since knowledge can be expressed in a more natural by using fuzzy sets, many engineering and decision problems can be greatly simplified.

Fuzzy set theory implements classes or groups of data with boundaries that are not sharply defined (i.e., fuzzy) [5]. Any methodology or theory implementing crisp definitions such as the set theory, arithmetic, and programming, may be fuzzified by generalising the concept of a crisp set to a fuzzy set with blurred boundaries. The benefit of extending crisp theory and analysis methods to fuzzy techniques is the strength in solving real-world problems, which inevitably entails some degree of imprecision and noise in the variables and parameters measured and processed in the application. Accordingly, linguistic variables are crucial for some fuzzy logic applications, where general terms such a large, medium, and small are used to group a range of numerical values. While similar to conventional quantification, fuzzy logic allows these stratified sets to overlap (e.g., a 60 kilogram woman may be placed in the large and medium categories, with varying degrees of belonging or membership to each group). Fuzzy set theory encompasses fuzzy logic, fuzzy arithmetic, fuzzy mathematical programming, fuzzy topology, fuzzy graph theory, and fuzzy data analysis, though the term fuzzy logic is often used to describe all of these [6].

Fuzzy logic emerged into the mainstream of information technology in the late 1980's and early 1990's. Fuzzy logic is a departure from classical Boolean logic, implementing soft linguistic variables in a continuous range of truth values which allows intermediate values to be defined between the conventional binary. It can often be considered a superset of Boolean or crisp logic in the way that the fuzzy set theory is a superset of conventional set theory. Since fuzzy logic can handle information in a systematic way, it is ideal to control nonlinear systems and to model complex systems where an inexact model exists, or for systems where ambiguity or vagueness is common. A typical fuzzy system consists of a basic rule, membership functions, and an inference procedure. Fuzzy Rules have the structure IF...THEN..., where both IF and THEN terms are natural language expressions of some fuzzy classes or their combinations. Fuzzy logic provides powerful computational techniques to manipulate these classes. So that, the input/output data of the HES discussed in this work use also this tool to develop it. In the next sections, it is going to be better described.

2.3 Evolutionary Computation

Evolutionary computation mimics the biological evolution with its ideas of natural selection and survival of the fittest to provide effective solutions for optimisation problems. The first approach to evolutionary computation was the Genetic Algorithm (GA) developed by John H. Holland in the 1960's [2].

The GA uses the concept of solution states encoded as binary-valued strings where a bit is analogous to a gene and the string is analogous to a chromosome. A set (i.e., population) of candidate solution states (i.e., chromosomes) is generated and evaluated. A fitness function is used to evaluate each of the solutions in the population. The chromosomes encoding better solutions are broken apart and recombined through the use of genetic operators such as crossover, mutation, and recombination to form new solutions (i.e., offspring) which are generally better or more fit than the previous iteration (i.e., generation). The process is repeated until an acceptable solution is found within specific time constraints [7]-[13]. Since this approach is significantly different from others, GA has been successfully applied to optimisation problems where other approaches failed.

GA has proven to be well suited to optimisation of specific nonlinear multivariable systems and it is being used in a variety of applications including scheduling, resource allocation, training ANN, and selecting rules for fuzzy systems. In this work, this tool is used to optimise the number of neurons in a hidden layer of the NNES [13][14]. The following sections describe its implementation.

3 Proposed System

The proposed architecture to the ES includes NNES, RBES and EES. The NNES has been used to implement the ES as an alternative manner to the RBES. ANN are made through a big number of units. These units own some properties like natural neurons. Therefore, every unit has several inputs, some excitatories and others inhibitories. Moreover, these units process its input and generate an output which is a function of the inputs. So, an ANN is characterised by the units (neurons), their connections (topology), and also by the algorithms used to change the connection weights (learning rules). Thus, these three aspects constitute the connectionist paradigm of the AI [15]. This ES implementation is called NNES. These systems are generally developed using a static network with feedforward topology trained by a backpropagation learning algorithm [16].

So, the basic network represents relations among concepts and connections and a set of examples refine the NNES. In this procedure, the algorithm modifies the connection weights and the network structure. Using this topology, it generates and/or eliminates connections that were not present in the basic rules. Besides, it can generate, occasionally, other concepts beyond those existing in the basic rules. Therefore, the system translates the basic rules into new ones that the expert was not able to extract. After the extraction, the basic rules suffer a treatment according to the variables applied to the network inputs, where they represent different types of concepts, as quantitative, linguistic, boolean or their combination [6]-[13][16]. Moreover, an ANN structural modification allows to determine the number of neurons of the hidden layer by using a GA. However the RBES depends strongly on formal logic as knowledge representation method. The developed system has two kinds of data: basic rules and example set. The basic rules are used to create an initial NNES which is refined with the examples. The refined NNES is then translated to a RBES from which explanations can be obtained. The proposed model is through fuzzy logic. The theory of fuzzy logic provides a good mathematical framework to represent imprecise knowledge as aimed in this work.

Finally, the EES of the HES is derived from RBES. It compares the answers given by the NNES and the RBES. When two answers are identical, the EES is triggered to give an explanation. Otherwise, it states the impossibility of reaching the goal and try to suggest how to achieve a suitable solution.

4 Methodology

The main idea is that, generally, the domain expert has difficulties in specifying all rules, mainly when imprecision is pervasive to the problem and fuzzy techniques need to be used. In this case, it is often hard to choose the membership function. Nevertheless, he is able to supply examples of real cases. So, the knowledge engineer use the rules that were supplied by the domain expert to implement a basic structure of a NNES. After, the NNES is refined by a training algorithm and available example set.

The KA task consists of extracting knowledge of the domain expert. In this work, the main goal is to minimise the intrinsic difficulties of the KA process. It is aimed, in a short time interval, to get all possible rules from the domain expert and also an example set. Moreover the rules should be improved to add a way to capture the uncertainties associated to human cognitive processes. The model proposed uses fuzzy logic. The theory of fuzzy logic provides a great mathematical framework to represent this kind of knowledge [6]-[13][16].

The next step is the implementation of the NNES which uses neurons representing concepts. The rules associating these concepts are used to establish the topology of the ANN, employing a graphic tool known as AND/OR graphs as aid to develop the basic structure of this system. In other words, AND/OR graphs, which represent concepts and connections, indicate the number of neurons in the input and output layers. They also show the existence of intermediate concepts and their connections which are translated into the intermediate layer of the NNES [7]-[13]. Besides, the NNES also foresees the possibility of different kinds of variables in its inputs, where they represent different types of concepts, as quantitative, linguistic, or boolean valued or their combination [6][16]. This is the way by which the basic NNES is obtained. In the following, it is going to be shown the mathematics model of the neuron, given by [6],

X(t) = n-dimensional input vector in the ANN or output of neuron exciting the neuron considered.

$$X(t) = [x_1(t), x_2(t), \cdots, x_i(t), \cdots, x_n(t)]^T \epsilon \Re^n \tag{1}$$

y(t) = o(t) = scalar output of each neuron and $o(t) \epsilon \Re^1$
N: nonlinear mapping function, $X \rightarrow O, x(t) \mid\!\rightarrow o(t)$, where:

$$X : Z^+ \epsilon \Re^n,\ O : Z^+ \epsilon \Re^1 \tag{2}$$

This mapping can be noted as N and so:

$$o(t) = N[X(t) \epsilon \Re^n] \epsilon \Re^1 \tag{3}$$

Mathematically, the neural nonlinear mapping function N can be divided into two parts: a function called confluence and a nonlinear activation operation [6]. Then confluence function © is the name given to a general function having as arguments the synaptic weights and inputs. A particular case widely used is the inner product. This mapping yields a scalar output $u(t)$ which is a measure of the similarity between the neural input vector $X(t)$ and the knowledge stored in the synaptic weight vector $W(t)$. So, $u(t) \epsilon \Re^1$ and $W(t)$ is given by,

$$W(t) = [w_0(t), \cdots, w_i(t), \cdots, w_n(t)]^T \epsilon \Re^{n+1} \tag{4}$$

Redefining $X(t)$ to include the bias $x_0(t)$ we have:

$$u(t) = X(t) \copyright W(t) \tag{5}$$

Refer to [7]-[13] in order to find the formulas applied to this stage. In short, for OR neurons, in (5), replacing the ©-operation by the T-operation, and the $\sum$-operation by the

$$u(t) = S_{i=1}^{n}[w_i(t)\ T\ x_i(t)] \epsilon [0,1] \tag{6}$$

Then for AND neurons, in (5), replacing the ©-operation by the operation of the algebraic product, and the $\sum$-operation by the T-operation, we get

$$u(t) = T_{i=1}^{n}[w_i(t) * x_i(t)] \epsilon [0,1] \tag{7}$$

$$o(t) = \Psi[u(t)] \epsilon [0,1] \tag{8}$$

Other stage of the HES is the refinement of the NNES. This is achieved through a learning algorithm using real examples as a training set. This learning algorithm allows structural changes of the network through inclusion and/or exclusion of neurons and/or connections. This approach leads to a localised knowledge representation where neurons represent concepts and connections represent relations among concepts [7]- [13].

After obtaining the basic NNES, the example set is used to validate its structure. In the worst case, the NNES does not represent the problem knowledge. Therefore, it becomes evident that the basic rules extracted from the domain expert are not enough, as expected. So, these same examples are used by the learning algorithm to refine the NNES. After its refinement, a new discussion is made within the domain expert to validate the changes implemented in its basic structure. Thus, a new example set is obtained to test the NNES again. When the results are satisfactory, it is supposed that the NNES represents the proposed goal.

Finally, a reverse process is followed towards the inferring of the *if-then* rules together with their membership degrees. So, a RBES is implemented. Following, the RBES works as the basis for the development of another system, the EES, that is supposed capable to explain why the NNES reached a conclusion.

5 Learning Algorithm

The learning algorithm developed is inspired on the classical backpropagation algorithm [4]. Nevertheless it presents some differences: optimisation of the hidden layer is supported by GA, incorporation of logic operators AND/OR in place of the weighted and, formation of the NNES by fuzzy logic. This learning algorithm changes not only the connection weights, but also the network structure. It generates and/or eliminates connections among those fuzzy basic rules given by the expert. Due to this, the number of neurons in the hidden layer must be changed. It has been suggested in the literature that such optimisation in the hidden layer can be accomplished using GA. [7]-[13].

The GA is used to optimise the hidden layer size and determine weights to be zeroed. This can be justified by the following main facts: it avoids local optimum and provides near-global optimisation solutions and they are easy to implement. Nevertheless, when it is applied to the hidden layer of an ANN, care must be taken in order to keep its maximum and minimum number of neurons. In fact, the use of too many neurons reduces the network generalisation capabilities and has got a slow learning phase. On the other hand, too few neurons might be unable to learn, with the desired precision. So, there is an optimal neuron number that must be placed in the hidden layer to avoid the problems mentioned above. Then, there is a trade-off to develop an efficient ANN with a short training cycle [7]-[13].

However, because of the difficulties in the analysis of min and/or max operation, the fuzzy ANN training, especially min-max ANN, does not appears to be approachable rigorously and systematically. Therefore, in practice, one tends to choose bounded-addition and multiplication to replace the min and max operations to eliminate the problem. Despite the fact that the modified ANN is readily trainable due to its analytical nature, it is functionally very different from the original one. In a sense, the lack of an appropriate analytical tool for the min and max operations greatly limits their applicability.

In [13][17], the authors have tried to develop a rigorous theory aiming to differentiate min-max functions by means of functional analysis, and derived the delta rule for training min-max ANN based on the differentiation theory of min-max functions. So, we applied to this backpropagation algorithm modified by [13] to the NNES.

6 Simulations

In this section, we will provide an example to show that the cognitive computing is effective to develop the HES. The case study illustrates the HES application to the problem of epilepsy diagnosis. The epilepsy data bank was supplied by physicians working, mainly, at the University Hospital of Federal University of Santa Catarina, Brazil. The data bank contained 39 symptoms and 4 diagnostic classes.

Let us give a min-max NNES with three layers, i.e., an input layer, an output layer, and a hidden layer. Since the range of values of the min-max NNES is often constrained in [1,- 1], the activation function for each neuron is chosen as the hyperbolic tangent. Below, some simulations are described. The NNES used has got: input layer = 6 neurons, output layer = 3 neurons, hidden layer = 6 neurons, Generation Number (G) = 10, Initial Population (P) = 8, Gauss Distribution, Crossover Rate (C) = 0.6, Mutation Rate (M) = 0.1, Learning Rate (α) = 0.1, Momentum (β) = 0.7, Tolerance (T) = 0.1, Maximum Epoches (ME) = 5, and Total Epoches (E) = 50. With this training data, 5 neurons were obtained for the hidden layer. As the number of most significant data of a pattern set was 4, this simulation shows that for a basic NNES with 6 neurons in the hidden layer, this number can be minimised to a number close to the most important data obtained during the knowledge extraction.

For the same data given above, except: ME = 50 and E = 500, the training data produced also 5 neurons in the hidden layer. Nevertheless it is observed that the quality of the network is improved. In this case, the value of Relative Variance Coefficient (RVC), e.g., RVC = Standard Deviation/Average Value, decreased, while the fitness increased.

In other simulation, we changed C to 0.8 and kept the other values. The number of neurons in the hidden layer was optimised to 4 neurons. It was observed that the standard deviation curve reached upper values with respect to previous simulations. So the dispersion of the variable (fitness) in relation to average of this variable (average of the fitnesses) increased. A great crossover number and mutations occurred in some generations.

Nevertheless, some more simulations were carried out changing, for example, M to 0.2 and keeping the other values used above. The final NNES achieved 4 neurons in the hidden layer and, changing P to 30 the NNES kept 4 neurons in the hidden layer. It was observed that a larger diversity of the initial chromosome population of a given population of individuals was brought about as by the change of P values as M values. Moreover we can see in [12][13] other simulations and other aspects for the problem approached here.

7 Conclusions

The complex systems associated to human activity are often poorly defined. Cognitive computing provides an effective and efficient way to analyse technological processes and human activities. In this work, a HES, including a RBES and a NNES, is discussed under the aspects of KA, where the treatment of imprecision is used to explain the reasoning to achieve a conclusion. Fuzzy sets and fuzzy logic can be used to express uncertain information in exact form. With this model, the method has proved, in the preliminary studies performed, very promising by providing an easier KA phase than that expected in KA performed with the use of symbolic techniques alone. The hybridism, on the other hand, allows to complement the NNES with explanation of reasoning facilities, that are difficult to achieve with a NNES in most cases.

The training of the fuzzy NNES was inspired by the classic backpropagation algorithm, with some alterations, as mentioned above.

Besides, it was observed that in the backward pass, the error propagation among the layers reached to the expected values. In this work, it was observed one of the limitations of a feedforward network with multiple layers which uses an algorithm similar to the backpropagation one. It requires a activation function that is nonlinear and differentiable. In this work, the hyperbolic tangent was used. However when the necessary operations in the backward pass were accomplished, a problem occurred. As the NNES is AND/OR, these functions are not differentiable, a priori. Nevertheless, it could be observed in [17] that this difficulty was attained through of a function called *lor* function.

Regarding to optimisation of the topology to be adopted for the fuzzy NNES, the aim was achieved. The optimisation of the hidden layer was supported by GA. To maximise the hidden layer, the adopted solution considered the following aspects: the values for crossover and mutation rates were chosen empirically, e.g., the 0.6 value for the crossover operator and 0.1 for the mutation operator. Using these values, the system reached a good performance. However during the mutation process, there was the creation of a population of chromosomes codifying a hidden layer too large to be an acceptable solution. To eliminate this, the peculiarities of the example were considered. In the minimization case, the following aspects were considered: for a given example set, it was possible to determine which values are more predominant. Then the minimum value of chromosomes generated by the selection process and genetic operators should be decreased to a maximum equal to the number of important data related to the example set.

Finally, after the optimised network is achieved, it is necessary to refine it. When the winner NNES was obtained, the number of neurons in the optimised hidden layer as the training NNES. In the next step, other test sets are applied to the NNES in order to refine it. It was observed that the most part of the sets of test patterns were recognised by the NNES. It is expected that the knowledge does not change according to domain chosen. Besides, it was also observed that the complex problems mentioned in this work were solved applying cognitive computing.

References

1. Johnson, R.C.: What is cognitive computing? Dr. Dobb's Journal (1993).
2. Holland, J.H.: Adaptation in Natural and Artificial Systems. MIT Press, Cambridge, MA (1975).
3. Hogan, J.M., Diederich, J: Random neural networks of biologically plausible connectivity. Complexity International, Vol. 2 (1996).
4. Rumelhart, D.E, Hinton, G.E., Willians, R.J.: Learning internal representations by error propagation. In: Rumelhart, D.E., McClelland, J.L., the PDP group.(eds.): Parallel Distributed Processing. Vol. 1. No. 2. MIT Press, Cambridge, Massachusetts (1987) 319–362.
5. Zadeh, L.A.: Fuzzy sets. Information and Control, Vol. 8 (1965) 338–354.
6. Gupta, M.M., Rao, D. H.: On the principles of fuzzy neural networks. Fuzzy Sets and Systems. vol. 61, no.1 (1994) 1–18.
7. Brasil, L. M., Azevedo, F.M., R. O., Garcia, Barreto, J.M.: Cooperation of symbolic and connectionist expert system techniques to overcome difficulties. in Proc. II Neural Networks Brazilian Congress. Curitiba, Brazil (1995) 177–182.
8. Brasil, L. M., Azevedo, F.M., R. O., Garcia, Barreto, J.M.: A Methodology for implementing hybrid expert systems. Proc. The IEEE Mediteranean Electrotechnical Conference, MELECON'96. Bary, Italy (1996) 661–664.
9. Brasil, L. M., Azevedo, F.M., Barreto, J.M.: Uma arquitetura híbrida para sistemas especialistas. Proc. III Neural Networks Brazilian Congress. Florianópolis, Brazil (1997) 167–172.
10. Brasil, L. M., Azevedo, F.M., Barreto, J.M.: Uma Arquitetura para Sistema Neuro-Fuzzy-Ga. Proc. of The III Congreso Chileno de Ingeniería Eléctrica, Universidad de La Frontera. Temuco, Chile (1997) 712–717.
11. Brasil, L. M., Azevedo, F.M., Barreto, J.M.: A hybrid expert architecture for medical diagnosis. Proc. 3th International Conference on Artificial Neural Networks and Genetic Algorithms, ICANNGA'97. Norwich, England (1997) 176–180.
12. Brasil, L. M., Azevedo, F.M., Barreto, J.M.: Learning algorithm for connectionist systems. Proc. XII Congreso Chileno de Ingeniería Eléctrica, Universidad de La Frontera. Temuco, Chile (1997) 697–702.
13. Brasil, L. M., Azevedo, F.M., Barreto, J.M., Noirhomme-Fraiture, M.: Training algorithm for neuro-fuzzy-ga systems. Proc. 16th IASTED International Conference on Applied Informatics, AI'98. Garmisch- Partenkirchen, Germany, February (1998) 697–702 (In Press).
14. Fogel, D.B.: Evolutionary Computation: Toward a New Philosophy of Machine Intelligence. IEEE Press, New York, USA (1995).
15. Azevedo, F.M. de: Contribution to the Study of Neural Networks in Dynamical Expert Systems. Ph.D. Thesis, Institut d'Informatique, FUNDP, Belgium (1993).
16. Mitra, S., Pal, S.K.: Logical operation based fuzzy MLP for classification and rule generation. Neural Networks. Vol. 7, No. 2 (1994) 25–29.
17. Zhang, X., Hang, C., Tan, S., Wang, P.Z.: The min-max function differentiation and training of fuzzy neural networks. IEEE Transactions on neural networks. No. 5 (1994) 1139–1150.

On Decision-Making in Strong Hybrid Evolutionary Algorithms

Carlos Cotta and José M. Troya

Dept. of Lenguajes y Ciencias de la Computación,
Complejo Politécnico (2.2.A.6), Campus de Teatinos,
University of Málaga, 29071-Málaga (SPAIN)
{ccottap, troya}@lcc.uma.es

Abstract. Any search algorithm is inherently limited if a broad enough range of problems is considered. Hybridization (use of problem-dependent knowledge) is the mechanism to overcome these limitations. This work discusses several ways to achieve such hybridization. It is shown that no efficient algorithmic tool exists to guide the design process. Therefore, two methodological heuristics are studied for that purpose: minimization of intra-forma fitness variance and the use of non-homogeneous representations. The effectiveness of these heuristics is supported by empirical results.

1 Introduction

The paradigm of evolutionary computation (EC) is currently one of the most active research areas. The simplicity of these techniques and the impressive success of their applications has motivated the interest of many researchers. Nowadays it is possible to find a great amount of scientific papers and practical interdisciplinary applications of evolutionary algorithms (EAs). Hence, they are considered robust and effective heuristics, easily applicable to any problem. In fact, a certain evaluation function is often the unique problem information they use.

However, there exist limitations to the quality of the results produced by any search algorithm. This fact was initially stated by Hart and Belew [5] and became popular with the so-called *No Free Lunch Theorem* (NFL theorem hereafter) by Wolpert and Macready [15]. In light of these results, it can be said that no "problem-independent robust solver" exists. In other words, it is necessary to incorporate knowledge of the problem under consideration in the search algorithm. Such augmented algorithms are called *hybrid algorithms.*

This work provides a generic approach to these techniques, focusing on decision-making considerations. The remainder of this paper is organized as follows: first, some implications of the NFL theorem are discussed (Sect. 2). Then, hybridization is introduced in Sect. 3. A specific form of hybridization (the so-called strong hybridization) is considered in Sect. 4. The complexity of the design process is studied in this section too. Subsequently, two design heuristics are described in Sect. 5. Finally, some conclusions are outlined in Sect. 6.

2 The NFL Theorem and Its Implications

The NFL theorem shows that the expected performance of any two search algorithms is the same for any metric defined over the histogram of (non repeated) sampled solutions when averaged over all problems. Therefore, if no a priori knowledge of the problem under consideration is available, any search algorithm (including random or enumerative search) yields the same expected results (see [15] for details).

Several arguments have been given questioning the validity of the NFL theorem in "real world". On the one hand, the use of metrics that do not consider resampled points is considered inadequate since most popular search algorithms are potentially retracing. While this is true, empirical results [2] show that the effects of resampling are mostly significant in advanced stages of the search in which, frequently, it has stagnated. Furthermore, these effects become negligible as the size of the problem grows.

On the other hand, the NFL theorem assumes for simplicity that the probability distribution $P(f)$ of problems is uniform. This is another controversial point. In fact, it seems logical to think that, in real world, some problems are more frequently found than others and hence $P(f)$ is not uniform. However, this does not alter the described scenario. Only having information about the problem being solved allows deciding which algorithm is more appropriate. Thus, using algorithm $\mathcal{A}$ instead of algorithm $\mathcal{B}$ for solving a problem f is justified if it is known that $\mathcal{A}$ outperforms $\mathcal{B}$ in domain ϕ and $f \in \phi$, i.e., f is one of the problems for which $\mathcal{A}$ is better than $\mathcal{B}$; without such information, any selection is arbitrary. In fact, it is difficult to ensure that algorithm $\mathcal{A}$ should be used for solving f if simply $P(f \in \phi) > P(f \notin \phi)$, since asymmetries and *minimax* distinctions (see Fig. 1) can exist in the performance curves of $\mathcal{A}$ and $\mathcal{B}$.

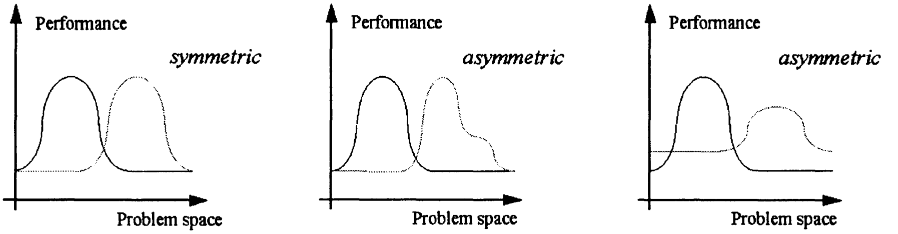


Fig. 1. Three scenarios in which *minimax* distinctions do not exist (left and middle) and do exist (right). Only the leftmost scenario is symmetrical.

Consider an algorithm $\mathcal{A}$ outperforming algorithm $\mathcal{B}$ over a problem domain $\phi_{\mathcal{A}}$ and achieving the same results over Ξ. It is possible to approximate the expected histogram of $\mathcal{A}$, $E(\vec{c}\ /m, \mathcal{A})$, as a weighted sum over $\phi_{\mathcal{A}}$, Ξ and $\phi_{\mathcal{B}}$ ($\phi_{\mathcal{B}} \equiv F \setminus (\phi_{\mathcal{A}} \cup \Xi)$, where F represents the whole problem space):

$$E(\vec{c}\ /m,\mathcal{A}) = \sum_{D=\{\phi_{\mathcal{A}},\Xi,\phi_{\mathcal{B}}\}} P(D)\cdot E(\vec{c}\ /D,m,\mathcal{A})\ , \qquad (1)$$

where $E(\vec{c}\ /D,m,\mathcal{A})$ is the expected histogram of points visited by $\mathcal{A}$ when applied to functions in D and $P(D)$ is its weight:

$$E(\vec{c}\ /D,m,\mathcal{A}) = \sum_{\vec{c},f\in D} \vec{c}\cdot P(\vec{c}\ /f,m,\mathcal{A})\cdot P(f/f\in D)\ . \qquad (2)$$

On the other hand, the expected histogram for a function f can be approximated as

$$E(\vec{c}\ /f,m,\mathcal{A}) = \sum_{D=\{\phi_{\mathcal{A}},\Xi,\phi_{\mathcal{B}}\}} P(f\in D)\cdot E(\vec{c}\ /D,m,\mathcal{A})\ . \qquad (3)$$

If no additional knowledge about f is available, then $P(f\in D) = P(D)$ and hence $E(\vec{c}\ /f,m,\mathcal{A}) = E(\vec{c}\ /m,\mathcal{A})$. Since, according to the NFL theorem, this quantity is independent of $\mathcal{A}$, there exists no information justifying the selection of a certain algorithm. Given that $P(f\in D)$ is known, algorithm $\mathcal{A}$ can be selected instead of algorithm $\mathcal{B}$ if $E(\vec{c}\ /f,m,\mathcal{A}) > E(\vec{c}\ /f,m,\mathcal{B})$. Expanding this expression using (3), and solving for $P(f\in\phi_{\mathcal{A}})$ yields

$$P(f\in\phi_{\mathcal{A}}) > P(f\in\phi_{\mathcal{B}})\frac{E(\vec{c}\ /\phi_{\mathcal{B}},m,\mathcal{B}) - E(\vec{c}\ /\phi_{\mathcal{B}},m,\mathcal{A})}{E(\vec{c}\ /\phi_{\mathcal{A}},m,\mathcal{A}) - E(\vec{c}\ /\phi_{\mathcal{A}},m,\mathcal{B})}\ . \qquad (4)$$

This equation proves that $P(f\in\phi_{\mathcal{A}}) > P(f\in\phi_{\mathcal{B}})$ does not suffice to justify the use of algorithm $\mathcal{A}$ instead of algorithm $\mathcal{B}$. Moreover, even if $P(f\in\phi_{\mathcal{A}}) > P(f\notin\phi_{\mathcal{A}})$ (obviously implying that $P(f\in\phi_{\mathcal{A}}) > P(f\in\phi_{\mathcal{B}})$) that selection may be unjustified. To see it, an analogous reasoning can be done just considering the regions $\phi_{\mathcal{A}}$ and $F\setminus\phi_{\mathcal{A}}$, yielding

$$P(f\in\phi_{\mathcal{A}}) > \frac{E(\vec{c}\ /F\setminus\phi_{\mathcal{A}},m,\mathcal{B}) - E(\vec{c}\ /F\setminus\phi_{\mathcal{A}},m,\mathcal{A})}{\Delta_{\phi_{\mathcal{A}},F\setminus\phi_{\mathcal{A}}}E(\vec{c}\ /m,\mathcal{A}) - \Delta_{\phi_{\mathcal{A}},F\setminus\phi_{\mathcal{A}}}E(\vec{c}\ /m,\mathcal{B})}\ , \qquad (5)$$

where $\Delta_{\phi,\psi}E(\vec{c}\ /m,\mathcal{X}) = E(\vec{c}\ /\phi,m,\mathcal{X}) - E(\vec{c}\ /\psi,m,\mathcal{X})$. Equation (5) allows deciding which algorithm is more appropriate according to $P(f\in\phi_{\mathcal{A}})$. Notice that if $E(\vec{c}\ /\phi_{\mathcal{A}},m,\mathcal{A}) = E(\vec{c}\ /F\setminus\phi_{\mathcal{A}},m,\mathcal{B})$ and $E(\vec{c}\ /F\setminus\phi_{\mathcal{A}},m,\mathcal{A}) = E(\vec{c}\ /\phi_{\mathcal{A}},m,\mathcal{B})$, then $\mathcal{A}$ should be chosen when $P(f\in\phi_{\mathcal{A}}) > 1/2$, i.e., when $P(f\in\phi_{\mathcal{A}}) > P(f\notin\phi_{\mathcal{A}})$. However, this should not be the situation in general.

To sum up, selecting a certain algorithm for the resolution of a problem f is justified if that algorithm is adequate for solving problems in a domain ϕ and $P(f\in\phi)$ is high enough. For the former requirement, it is necessary that the algorithm use knowledge about domain ϕ in order to intelligently solve such problems. To ensure the latter requirement, the problem being solved must be studied and their major features have to be checked against those of ϕ. This aspect falls beyond the scope of this work, which will focus on the first requirement.

3 Hybridization. A Dichotomy

The NFL theorem places us in a playing field in which no algorithm is better on average than any other one. Therefore, it is necessary to use algorithms whose requirements with respect to the search space match the current problem. This does not imply that using EAs is inappropriate, but simply emphasizes that any given EA (e.g., a generational genetic algorithm using binary encoding, one-point crossover and bit-flip mutation) will not successfully solve every problem. Nevertheless, it is possible to augment the basic EC-skeleton with problem-domain knowledge, so as to specialize the algorithm. These algorithms (based in the architecture of EAs and using problem-domain knowledge) are called *hybrid evolutionary algorithms*. Very different ways of achieving hybridization are possible. Prior to gain a better understanding of hybrid EAs, it is necessary to establish a systematic classification of the several ways of introducing knowledge in them. Two major mechanisms can be considered.

On the one hand, knowledge can be placed into the core of the algorithm, affecting its internal components. Specifically, a representation of problem solutions allowing the extraction of their most important features, and reproductive operators working on that representation are required. These two aspects are closely related and must be carefully selected to obtain adequate performance. We will term any hybrid algorithm resulting from such an augmentation a *strong hybrid algorithm*, thus reflecting the tight coupling that exists between the basic model and the included knowledge.

On the other hand, it is possible to combine algorithms performing different searches, thus resulting in a *weak hybrid algorithm*. This term tries to capture the fact of hybridization taking place in a higher level and through a well-defined interface, therefore allowing algorithms to retain their identities. This terminology is consistent with the classification of problem-solving strategies in artificial intelligence as *strong* and *weak* methods [8].

We defer the study of weak hybridization to a further work for two reasons: first, it is a substantial topic for itself. Second, it only makes sense to consider weak hybridization in the context of knowledge-augmented algorithms (in fact, strong hybrids). Otherwise, the resulting algorithm would not be guaranteed to be better than random search.

4 Strong Hybridization

As mentioned above, a requirement for an evolutionary algorithm to be effective in a problem domain is the use of both appropriate representation and operators. Clearly, such ideas must be previously formalized.

4.1 Representations and Operators

Suppose an algorithm $\mathcal{A}$ using a representation function ρ (expressed as $\mathcal{A}(\rho)$ below). The idea of appropriate representation is defined in [13] as follows:

Definition 1 (appropriate representation). *A representation function ρ is appropriate for algorithm $\mathcal{A}$ over a problem domain ϕ if the performance of $\mathcal{A}(\rho)$ is better when averaged over ϕ rather than over the remaining problems, i.e., $E(\vec{c}\ /\phi, m, \mathcal{A}(\rho)) > E(\vec{c}\ /F \setminus \phi, m, \mathcal{A}(\rho))$.*

This definition allows comparing two algorithms using the same representation. Algorithm $\mathcal{A}$ is better than algorithm $\mathcal{B}$ for domain ϕ using representation ρ if, and only if, $E(\vec{c}\ /\phi, m, \mathcal{A}(\rho)) > E(\vec{c}\ /F \setminus \phi, m, \mathcal{B}(\rho))$. Notice that Definition 1 does not consider the effects of the operators the algorithm uses. However, it is well known that both elements are highly related and must be correctly tuned with each other. Therefore, a more adequate definition must include both elements:

Definition 2 (problem view). *Let f be a problem and let $\mathcal{A}(\rho, \omega)$ be an algorithm using representation function ρ and operator ω. The view of f with respect to $\mathcal{A}$ is the tuple $\mathcal{V}(f, \rho, \omega)$.*

The above definition essentially agrees with the notion of landscape [6], implicitly subsuming the abstractions of fitness space and partial order over it. It is easy to extend Definition 2 to consider problem domains. The concept of appropriate view can then be defined as follows:

Definition 3 (appropriate view). *An algorithm $\mathcal{A}(\rho, \omega)$ has an appropriate view of a problem domain ϕ if its performance is better than random search when averaged over all problems in that domain, i.e., $E(\vec{c}\ /\phi, m, \mathcal{A}(\rho, \omega)) > E(\vec{c}\ /\phi, m, \text{random}(\rho))$.*

Arguably, this definition sets a small lower bound when an algorithm using a certain representation and operators is evaluated. However, it is a first step that is specifically interesting for evolutionary algorithms. Consider that, grosso modo, they perform a guided random search. If the quality of this search does not improve pure random search, the guidance mechanism is incorrect or, at least, useless.

Notice also that Definitions 1 and 3 are neither equivalent nor special cases of each other. Although there exist situations in which they agree, alternative scenarios can be imagined in which an algorithm performs its best in a given domain but, nevertheless, it is outperformed by random search. These situations are not pathological cases. As an example, consider the performance curves shown in Fig. 2.

Each horizontal segment corresponds to the performance of an algorithm averaged over a domain $\phi_i, 0 \leq i \leq 4$, where the subscript i represents the number of *peaks* [6] in the problem view. It can be seen that hill climbing is better in ϕ_0 (corresponding to views without any acute peak and hence dominated by *mesas*) than in any other domain. However, random search is much better in that domain. In this scenario, it does not seem reasonable to claim that hill climbing has an appropriate view of ϕ_0. Notice that if performance is normalized considering the results of random search as a baseline, then ϕ_3 turns to be the region in which hill climbing is comparatively better.

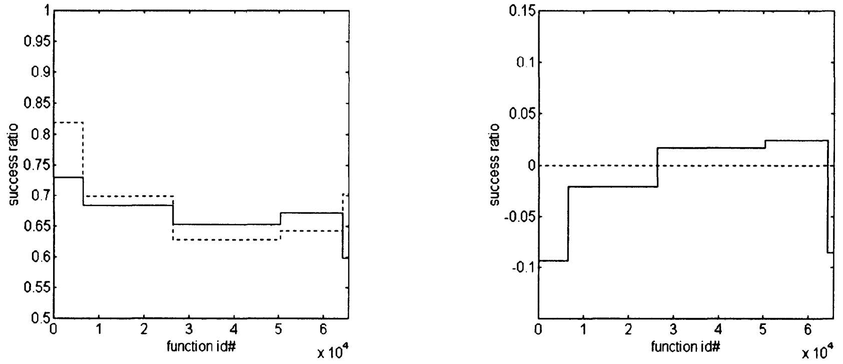


Fig. 2. Raw (left) and normalized (right) performance of hill climbing (solid line) and random search (dotted line) across the set of functions $f : D \rightarrow I$, with $|D| = 8$ and $|I| = 4$. Each algorithm visits 4 points of the search space.

4.2 The Hardness of Selecting a View

Selecting an appropriate view for a certain algorithm $\mathcal{A}$ and function f implies the definition of a representation and operators that optimize the output of $\mathcal{A}$ with respect to a performance indicator Ψ. Unfortunately, such optimization is hard to solve. To show it, this process will be characterized as the decision version of a combinatorial optimization problem as in [5].

Definition 4 (FIND-VIEW).
INSTANCE: a tuple (n, Ψ, f, λ) *where* n *is the size of the representation (assuming binary encoding without loss of generality),* Ψ *is the performance indicator,* f *is the target function and* λ *is a threshold.*
QUESTION: Does there exist a polynomial-length sequence $\langle x_i \rangle$ $(x_i \in \mathcal{B}_n)$*, such that* $\Psi(\langle x_i \rangle) > \lambda$*?*

Assume that both f and Ψ can be evaluated in $\mathbf{TIME}(n^r)$, which seems reasonable. This problem is NP-hard: it can be reduced to SAT as shown below.

Proposition 1 (Complexity of FIND-VIEW). *FIND-VIEW is NP-hard.*

Proof. First, notice that FIND-VIEW is in NP. A polynomial-length sequence can be guessed in polynomial time by a nondeterministic machine and then evaluated in $\mathbf{TIME}(n^s)$. Thus, FIND-VIEW can be solved in $\mathbf{NTIME}(n^k)$.

Next, SAT is reduced to FIND-VIEW. An instance of SAT is composed of a tuple (η, φ) where η is a set of variables and φ is a set of clauses in conjunctive normal form. A reduction from SAT to FIND-VIEW can be performed using a Turing machine that on input $u = (\eta, \varphi)$ outputs $R(u) = (|\eta|, \Psi, f_\varphi, 0)$, where Ψ is the *max* function (commonly used to evaluate the output of an algorithm) and f_φ is a function that evaluates φ assigning the value of the i-th bit of x to the i-th variable of η. Since this reduction only uses space to store $|\eta|$ this is a log-space reduction. Now, observe that if φ is satisfiable then there exists an element x_i for which $f_\varphi(x_i) > 0$, and hence a sequence for which $\Psi(\langle x_i \rangle) > 0$. Conversely, if for a certain sequence $\Psi(\langle x_i \rangle) > 0$, then there exists an element x_i that represents a truth assignment satisfying φ. Thus, $w \in \texttt{SAT} \Longleftrightarrow R(u) \in \texttt{FIND} - \texttt{VIEW}$.

□

Notice that, even knowing that there exist appropriate views for a problem, the function version on the problem is still in FNP. This result implies that, unless P = NP, there exists no other option than relying on design heuristics to select a view for a given algorithm.

5 Heuristics to Select a View

It would be very hard to give a complete overview of the plethora of methodological heuristics (many of them settled on a strongly formal basis) available to aid the design of an evolutionary algorithm: *Walsh-coefficients analysis* [4], *Forma analysis* [11], etc. We will focus on two promising mechanisms: the use of non-homogeneous representations and the minimization of fitness variance of formae.

5.1 Using Non-Homegeneous Representations

The rôle of representation is to provide a mechanism for handling abstract solutions to a problem in a computer. There exists several properties that, traditionally, have been imposed to representations. On the one hand, every solution should have a corresponding representation. On the other hand, nothing but feasible solutions should be represented. Therefore, it seems that a bijective mapping between feasible solutions and representations is the perfect option. The rationale for this is twofold: not to exclude any solution that could be optimal and not to waste computational efforts in storing and handling unfeasible solutions.

Some arguments can be given against this idea. Consider that enough problem-knowledge can be available to discard some (feasible) solutions which are known to be non-optimal. These discarded solutions are "unfeasible" de facto. Therefore a representation using this knowledge should not include them. Moreover, the representation can exploit this knowledge to concentrate the search in promising regions by means of controlled *degeneracy* (more than one representation for a solution). Representations with these features are called *non-homogeneous.*

Non-homogeneous representations can be constructed by means of a transformation of the original representation as shown in Fig. 3 (left). These transformations can be done in different ways: using a construction heuristic [14], by means of a hill-climber [10], etc. As an example, Fig. 3 (right) shows two fitness distributions. They correspond to a bijective and a transformed representation of an integer multidimensional knapsack problem (0-1 MKP) taken from the OR-Library [1]. This transformation has been done using a greedy heuristic [3]. It can be seen that the fitness distribution of the latter representation has a higher mean and a smaller variance than the distribution of the former.

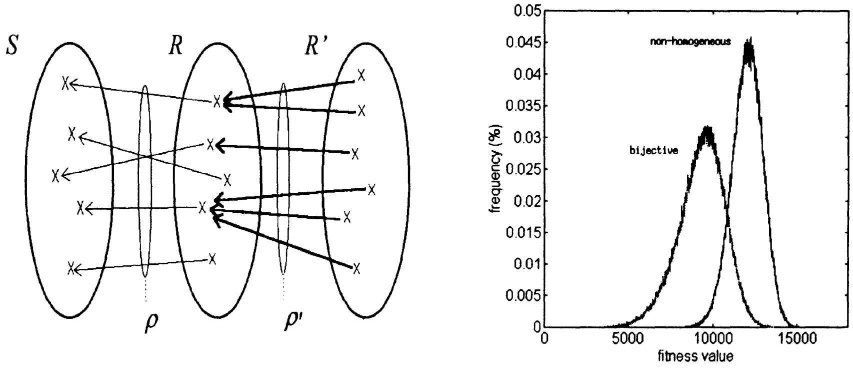


Fig. 3. A bijective representation function (ρ) and a non-homogeneous representation (ρ') are shown at left. The fitness distributions of these two representations is shown at right.

5.2 Minimizing Fitness Variance of Formae

The use of a non-homogeneous representation does not suffice for itself to ensure an effective search. It just ensures than a random guesser will perform better using the transformed rather than the original representation. To turn this choice of representation really useful, the operators used in the algorithm must manipulate relevant features of represented solutions. These features (formae) define the quality of the problem view.

Since EAs use fitness as a guidance mechanism, the formae manipulated by the operators must carry significant information in this sense. Again, there exist many heuristic indicators of this property: high fitness correlation from parents to offspring, null high-order Walsh coefficients, etc. Radcliffe and Surry [12] have proposed measuring the fitness variance of formae. The idea is that low variance induces fewer sampling errors and thus makes the algorithm more predictable. It must be stressed that all these considerations make sense only if the operators are *transmitting* [10] or at most induce very limited implicit mutation. Otherwise, factors such as forma disruption would become dominant.

Several experiments have been carried out to measure the correlation of low fitness variance with good performance. Results are shown in Fig. 4 and Table 1. We have compared two genetic algorithms using the transformed representation with a *Random Respectful Recombination* operator [9], and a direct representation with one-point crossover and a penalty function used in [7]. As it can be seen, these results are consistent with the predictions of the fitness variance indicator; the hybrid algorithm is much more successful at finding the optimal solution. To extend these results, a similar analysis has been carried out on a different problem, the TSP. In this case, the non-homogeneous representation has been tested against an undirected-edge representation with an edge-recombination operator. Again, the non-homogeneous representation is better, providing more accurate solutions as shown in Table 1.

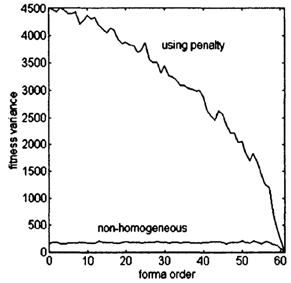


Fig. 4. Fitness variance of formae in two representations of the 0-1 MKP. The results correspond to the sento2 problem.

Table 1. Empirical results for the 0-1 MKP (100 runs) and for the TSP (10 runs). It is shown the success ratio and the deviation from the optimal solution respectively.

0-1 MKP	with penalty	non-homogeneous	TSP	undirected-edge	non-homogeneous
Knap50	1%	46%	kroA100	8.17%	3.79%
Sento1	5%	75%	kroB100	6.70%	2.66%
Sento2	2%	39%	kroC100	9.07%	3.84%
Weing7	0%	40%	kroD100	7.90%	4.58%
Weing8	6%	29%	kroE100	7.87%	4.55%

6 Concluding Remarks

The limitations of EAs have been exposed in this work. According to the NFL theorem, no search algorithm is better than random search on average. Consequently, EAs must be adapted by means of problem-dependent knowledge to perform an effective search in different domains. The necessity of incorporating such knowledge is satisfied by the mechanism of hybridization. In this sense, two possibilities have been considered: strong hybrid methods (including problem-knowledge in the core of the algorithm) and weak hybrid models (combinations of different search algorithms).

A critical point in strong hybrid EAs is the choice of representation and operators, which can be considered in fact as the two sides of the same coin. Unfortunately, it has been shown that this choice poses in general a hard problem. Therefore, we are confined to heuristics for guiding the decision-making process of an EA design. Clearly, an extensive study of all these heuristics is unaffordable in this paper. Thus, we have only described two mechanisms we believe to be promising since they can be empirically measured: the use of non-homogeneous representations and operators handling low-variance formae. These heuristics have been shown to be effective on a number of benchmark problems.

Many future lines of work are open. For example, it must be determined the validity of these heuristics in presence of weak hybridization, i.e., when the objective is to establish effective cooperation rather than individually optimize each single algorithm. In this context, weak hybrid models offer a vast universe of possibilities.

References

1. J.E. Beasley. Or-library: Distributing test problems by electronic mail. *Journal of the Operational Research Society*, 41(11):1069–1072, 1990.
2. C. Cotta. On resampling in nature-inspired heuristics. In V. Botti, editor, *Proceedings of the Seventh Conference of the Spanish Association for Artificial Intelligence*, pages 145–154, 1997. In Spanish.
3. C. Cotta and J.M. Troya. A hybrid genetic algorithm for the 0-1 multiple knapsack problem. In G.D. Smith, N.C. Steele, and R.F. Albrecht, editors, *Artificial Neural Networks and Genetic Algorithms: Proceedings of the Third International Conference*, pages 251–255, Wien, 1997. Springer-Verlag.
4. D.E. Goldberg. Genetic algorithms and walsh functions: Parts i and ii. *Complex Systems*, 3:129–171, 1989.
5. W.E. Hart and R.K. Belew. Optimizing an arbitrary function is hard for the genetic algorithm. In R.K. Belew and L.B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 190–195, San Mateo, CA, 1991. Morgan Kaufmann.
6. T.C. Jones. *Evolutionary Algorithms, Fitness Landscapes and Search.* PhD thesis, University of New Mexico, 1995.
7. S. Khuri, T. Baeck, and J. Heitkoetter. The zero/one multiple knapsack problem and genetic algorithms. In E. Deaton, D. Oppenheim, J. Urban, and H. Berghel, editors, *Proceedings of the 1994 ACM Symposium on Applied Computation*, pages 188–193. ACM Press, 1994.
8. Z. Michalewicz. A hierarchy of evolution programs: An experimental study. *Evolutionary Computation*, 1(1):51–76, 1993.
9. N.J. Radcliffe. Forma analysis and random respectful recombination. In R.K. Belew and L.B. Booker, editors, *Proceedings of the Fourth International Conference on Genetic Algorithms*, pages 222–229, San Mateo, CA, 1991. Morgan Kaufmann.
10. N.J. Radcliffe. The algebra of genetic algorithms. *Annals of Maths and Artificial Intelligence*, 10:339–384, 1994.
11. N.J. Radcliffe. Equivalence class analysis of genetic algorithms. *Complex Systems*, 5:183–205, 1994.
12. N.J. Radcliffe and P.D. Surry. Fitness variance of formae and performance prediction. In L. Whitley and M. Vose, editors, *Foundations of Genetic Algorithms 3*, pages 339–384. Morgan Kaufmann, 1994.
13. N.J. Radcliffe and P.D. Surry. Fundamental limitations of search algorithms: Evolutionary computing in perspective. In J. Van Leeuwen, editor, *Computer Science Today: Recent Trends and Developments*, volume 1000 of *Lecture Notes in Computer Science*, pages 255–291. Springer-Verlag, 1995.
14. R.H. Storer, S.D. Wu, and R. Vaccari. New search spaces for sequencing problems with application to job-shop scheduling. *Management Science*, 38:1495–1509, 1992.
15. D.H. Wolpert and W.G. Macready. No free lunch theorems for search. Technical Report SFI-TR-95-02-010, Santa Fe Institute, 1995.

Generalized Predictive Control Using Genetic Algorithms (GAGPC). An Application to Control of a Non-linear Process with Model Uncertainty*

Xavier Blasco, Miguel Martínez, Juan Senent, and Javier Sanchis

Department of System Engineering and Control
Universidad Politécnica de Valencia
46071 Valencia SPAIN
xblasco@isa.upv.es, mmiranzo@isa.upv.es, juanse@isa.upv.es, jsanchis@isa.upv.es
Fax : + 34 96 3879579 Tel : + 34 96 3877571

Abstract. Predictive Control is one of the most powerful techniques in process control, but its application in non-linear processes is challenging. This is basically because the optimization method commonly used limits the kind of functions which can be minimized. The aim of this work is to show how the combination of Genetic Algorithms (GA) and Generalized Predictive Control (GPC), what we call GAGPC, can be applied to non-linear process control. This paper also shows GAGPC performance when controlling non-linear processes with model uncertanties. Success in this area will open the door to using GAGPC for a better control of industrial processes.

1 Motivation and objectives

One limitation that should be overcome in a control process problem is the existence of non-linearities. They appear in different forms in almost all industrial processes, – for example, actuator non-linearities such as: saturations, dead-zones, backlash – or non-linear processes. Another important problem is model imperfection, usually the mathematical model cannot exactly reproduce the process behaviour.

Predictive control has demonstrated excellent performances in both theoretical studies and industrial applications ([2], [8], [3]). Even so, their application in the control of non-linear processes is complicated. This is basically because of the optimization method that has come to be used in these controllers. The proposed solution in this paper is based on the use of a very powerful optimizer: Genetic Algorithms ([4], [5]). The combination of Generalized Predictive Control GPC ([6], [7]), and Genetic Algorithms (GA) is what we call GAGPC.

In [10], [1] and [9] GAGPC controllers and their application in the control of processes with non-linear actuators are described in more detail. This work

* Supported by the Spanish Government Commission CICYT project TAP96-1090-C04-02.

shows how this technique is applied to non-linear processes, even with modelling errors, and remarks on how to use it in real-time control.

The work is divided in several parts that cover the following points:

- A brief description of the Generalized Predictive Control using Genetic Algorithms (GAGPC).
- GAGPC application to real-time control.
- GAGPC in non-linear process control with model errors.

2 Generalized Predictive Control Using Genetic Algorithms (GAGPC)

The predictive control used is the Generalized Predictive Controller (GPC) with the following essential characteristics:

- It requires a process model and a disturbance model to obtain output predictions (figure 1).

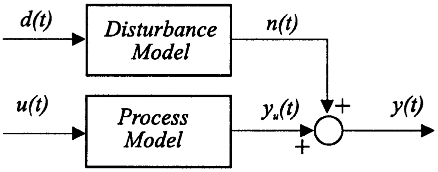


Fig. 1. Model to obtain predictions.

In the original formulation of the GPC, the CARIMA model is used:

$$y(t) = y_u(t) + n(t) \tag{1}$$

$$y(t) = \frac{B(z^{-1})}{A(z^{-1})}u(t-1) + \frac{T(z^{-1})}{\Delta A(z^{-1})}d(t) \tag{2}$$

From which the output predictions in time '$t + i$', with the available information until time 't', are obtained, $y(t + i|t)$.

- The control action is obtained from minimization of the following cost index:

$$J(u) = \sum_{i=N_1}^{N_2} \alpha_i[y(t+i|t) - w(t+i)]^2 + \sum_{j=1}^{N_u} \lambda_i[\Delta u(t+j-1)]^2 \tag{3}$$

Where $w(t)$ is the reference and $u = [u(t), u(t+1), \ldots, u(t+N_u)]$. The result is a control sequence that minimizes the cost index, $\hat{u} = [\hat{u}(t), \ldots, \hat{u}(t+N_u)]$.

- The control action applied each time is the first of the optimal control sequence, that is, $\hat{u}(t)$. In each sampling this sequence is recomputed, but only the first element is applied (this is known as *Receding Horizon*).

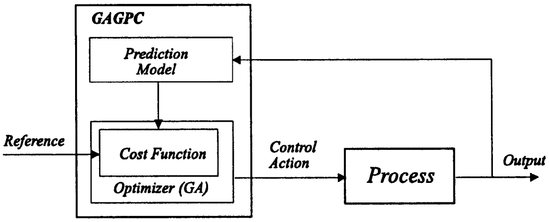


Fig. 2. Predictive control loop.

Figure 2 shows the structure of a predictive control loop.

As said before, the optimizer used in the GAGPC is a Genetic Algorithm. Genetic Algorithms are optimization technique based on the laws of species evolution. With each generation, a species evolves spreading to adapt better to their environment.

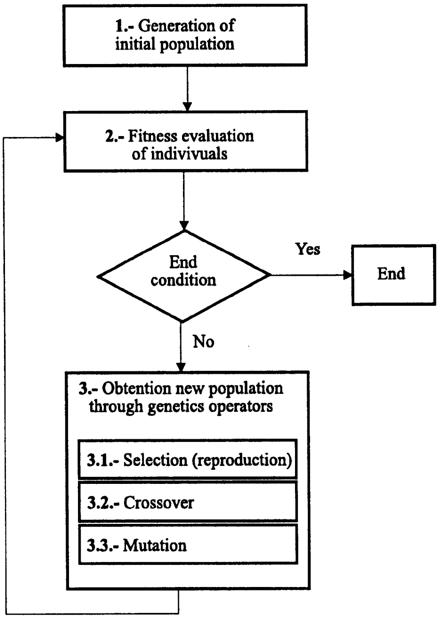

1.- Generation of initial population
2.- Fitness evaluation of indivivuals
End condition
Yes
End
No
3.- Obtention new population through genetics operators
3.1.- Selection (reproduction)
3.2.- Crossover
3.3.- Mutation

Fig. 3. Genetic Algorithm flow diagram.

To apply this idea to an optimization problem, an initial population (first generation) is composed of a set of points in the search space. All the searching space points have a chromosome type assigned, this is, a different string for each position in the searching space. Binary codification is usually selected to make chromosomes. Starting from this initial population, a genetic operator set is applied to obtain a new population (new generation). The results of genetic operator application depend on the cost index value of each individual. For example, if it is a maximization problem, the individuals with a bigger cost index value will have more probabilities of being selected for the following generation and participating in the crossing and mutation processes.

Figure 3 shows the Genetic Algorithm flow diagram used in this work – known as Simple Genetic Algorithm (SGA). Chromosomes strings are binary and the code is standard binary.

The GA characteristics set is:

- A *Ranking* operation is applied before the *selection* operation to avoid premature convergence.
- The *Selection* operator is Stochastic Universal Sampling (SUS).
- The *Crossover* operator is the Single Point Crossover, with a crossover probability of $Pc = 0.7$.
- *Mutation* operator is applied with a probability $P_m = 0.7/L$ for each bit, where L is the chromosome length.

	Sampling time	Process	Discrete model	Poles (p_i) zeros (z_i)
1	1-90	$\frac{1}{1+10s+40s^2}$	$\frac{0.0115z^{-1}+0.0106z^{-2}}{1-1.7567z^{-1}+0.7788z^{-2}}$	$p_{1,2} = 0.8784 \pm 0.0853i$, $z_1 = -0.92$
2	91-170	$\frac{e^{-2.7s}}{1+10s+40s^2}$	$\frac{0.0011z^{-3}+0.0159z^{-4}+0.0051z^{-5}}{1-1.7567z^{-1}+0.7788z^{-2}}$	$p_{1,2} = 0.8784 \pm 0.0853i$, $p_{3,4,5} = 0$, $z_1 = -0.3248$, $z_2 = -14.1858$
3	171-250	$\frac{e^{-2.7s}}{1+10s}$	$\frac{0.0296z^{-3}+0.0656z^{-4}}{1-0.9048z^{-1}}$	$p_1 = 0.9048$, $p_{2,3,4} = 0$, $z_1 = -2.2199$
4	251-330	$\frac{1}{1+10s}$	$\frac{0.0952z^{-1}}{1-0.9048z^{-1}}$	$p_1 = 0.9048$
5	331-400	$\frac{1}{10s(1+2.5s)}$	$\frac{0.0176z^{-1}+0.0154z^{-2}}{1-1.6703z^{-1}+0.6703z^{-2}}$	$p_1 = 1$, $p_2 = 0.6703$, $z_1 = -0.8753$

Table 1. Process and transfer function model.

The GAGPC controller allows a very flexible cost function, and there are no limitations in the model or the index type used to minimize it. These characteristics enable application to non-linear processes and could therefore solve many industrial processes control problems. It is necessary to mention that this control type has the inconvenience of a noticeable computational cost, and this could affect its application to real-time control. Nevertheless, the controller has some qualities which are useful for real-time control, and these aspects are studied below.

3 GAGPC application in real-time control

The cost function minimization of the predictive controller with a Genetic Algorithm means that the time calculation of each control action is noticeable. Indeed, with a GA it is necessary to evaluate a cost function for each individual, as well as applying the genetic operators – and this takes time. But GAs have a feature that facilitates their application in real-time. They offer a solution in any instant (once the first evaluation of the cost function for the first generation has been carried out).

To show this characteristic, GAGPC control is tested with a process that changes with time. Time restrictions are simulated by limiting the number of generations calculated. This is the equivalent of requiring a solution (control action) at different times.

The different process transfer functions are indicated in table 1, a one second sampling period is used to obtain the prediction models (discrete ones). These different transfer functions enable the analysis of the behaviour of different types of systems. Model changes are previously known in the generation of predictions, that is to say, for the calculation that the controller carries out, it knows all the transfer functions and when the changes will take place.

Parameters selected in the controller's calculation are:

- Prediction horizon $N_1 = 1$, $N_2 = 10$.
- Control horizon $N_u = 1$.
- Control action weight $\lambda_i = 0$.
- Number of individuals by generation $N_{ind} = 20$.
- Chromosome length of 16-bits (binary code).
- Searching space $[-200\ 200]$ (minimum control action and maximum).

The reference applied is indicated in table 2. The number of generations has been limited at 1 (*Sim1*), 5 (*Sim2*), 10 (*Sim3*) and 15 (*Sim4*) for the calculation of each control action. The results are shown in the figure 4.

As can be observed in figure 4, answers improve when there is more time for calculation (more generations are avoided), but in all cases, the process is under control even with very little time for calculation.

Value	20	0	20	0	20	0	20	0	20	0
Time	1-50	51-90	91-130	131-170	171-210	211-250	251-290	291-330	331-370	371-400

Table 2. Reference.

4 Non-linear process control with GAGPC

This section shows how the GAGPC is applied to a non-linear process. For this purpose, a process model and a disturbance model is needed to obtain predictions.

Genetic Algorithms can be applied to any type of model. These models must simply give the future output with the future reference, and the past input/output information. GPC control works with a CARIMA model (linear) but it does not work satisfactorily with a non-linear process. However, the GAGPC can use non-linear models and obtain good performances.

The next example shows GAGPC control of a process with the following equations:

$$\begin{aligned} I\ddot{q}_1(t) + Mgl\sin(q_1(t)) + K(q_1(t) - q_2(t)) &= 0 \\ J\ddot{q}_2(t) + B\dot{q}_2(t) - K(q_1(t) - q_2(t)) &= u(t) \end{aligned}$$

Where:

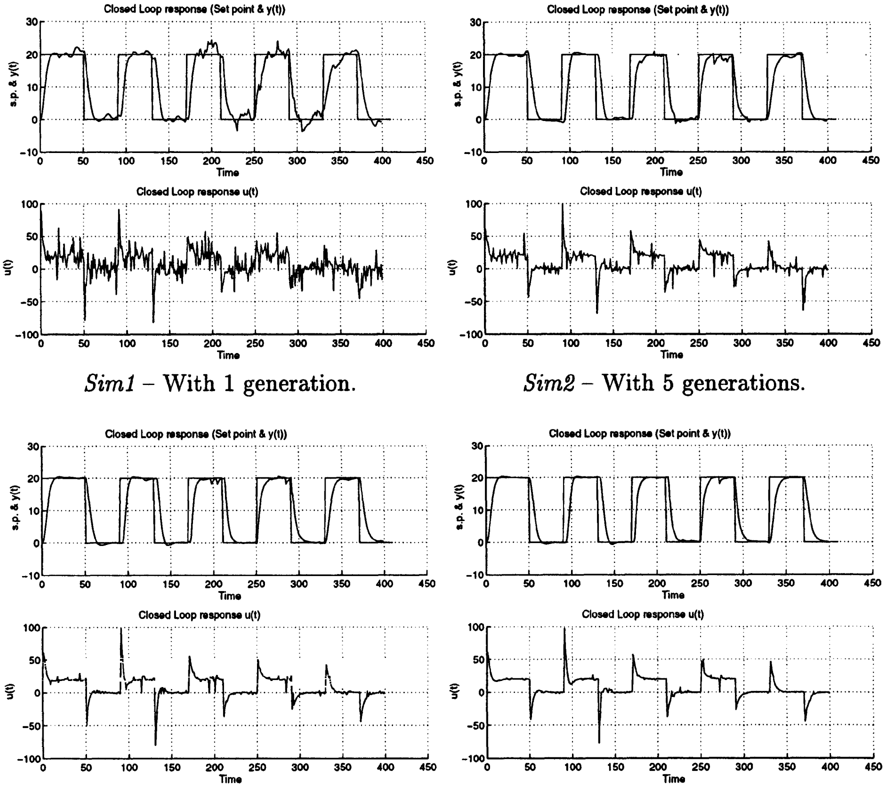
- $q_1(t)$ and $q_2(t)$ (and their first derivatives $\ddot{q}_1(t)$ and $\ddot{q}_2(t)$) are measurable system variables.
- $u(t)$ is the input to the system.
- $I = 0.032$, $J = 0.004$, $B = 0.007$, $Mgl = 0.8$ and $K = 7.13$ are system parameters.

A state space representation could be obtained taking as state variables: $x_1(t) = q_1(t)$, $x_2(t) = \dot{q}_1(t)$, $x_3(t) = q_2(t)$ and $x_4(t) = \dot{q}_2(t)$.

$$\begin{aligned} \dot{x}_1(t) &= x_2(t) \\ \dot{x}_2(t) &= -M_1\sin(x_1(t)) - K_1(x_1(t) - x_3(t)) \\ \dot{x}_3(t) &= x_4(t) \\ \dot{x}_4(t) &= -B_1x_4(t) + K_2(x_1(t) - x_3(t)) + u(t)/J \\ y_u(t) &= x_1(t) \end{aligned}$$

Where:

$$M_1 = \frac{Mgl}{I}\,,\, K_1 = \frac{K}{I}$$
$$B_1 = \frac{B}{J}\,,\, K_2 = \frac{K}{J}$$



Sim1 – With 1 generation. *Sim2* – With 5 generations.

Sim3 – With 10 generations. *Sim4* – With 15 generations.

Fig. 4. Results with different number of generations for the calculation of each control action.

The model for predictions (a discrete one) is obtained from this continuous model by discretizing it with Euler's method and a sampling period $T = 0.01 sec.$:

$$\begin{aligned}
x_1(k+1) &= x_1(k) + Tx_2(k) \\
x_2(k+1) &= x_2(k) - TM_1 \sin(x_1(k)) + TK_1(x_1(k) - x_3(k)) \\
x_3(k+1) &= x_3(k) + Tx_4(k) \\
X_4(k+1) &= x_4(k) - TB_1x_4(k) + TK_2(x_1(k) - x_3(k)) + Tu(k)/J \\
y_u(k) &= x_1(k)
\end{aligned}$$

Obviously, the quality of the model depends on the sampling period. The shorter it is, the better the model will be, and therefore, the predictions will be better. To show the model quality, a simulation with the model has been carried out and it is compared with the output process. Figure 5 shows the result, and an appreciable difference between the two outputs can be seen.

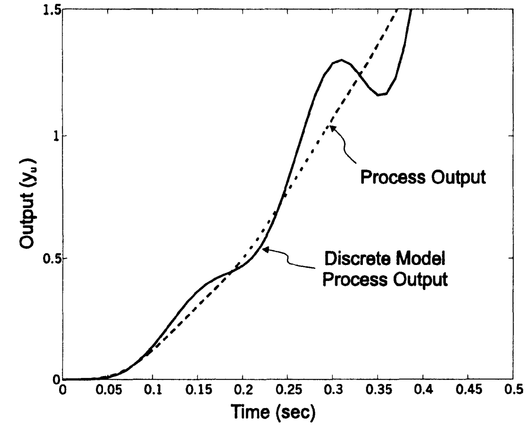


Fig. 5. Process and discrete model outputs for $u(t) = 1$.

The following disturbance model is used:

$$n(t) = \frac{1}{\Delta}\xi(t) \tag{4}$$

The GAGPC parameters are: $N_u = 1$, $N_1 = 1$, $N_2 = 10$, $\lambda_i = 0$, $\alpha_j = 1$, 40 individuals per generation, 20 generations, 16-bit chromosomes and limiting control action between $[-20, +20]$. All state values from the previous instant (all measurable variables) are passed to the model for prediction. The results are shown in figure 6.

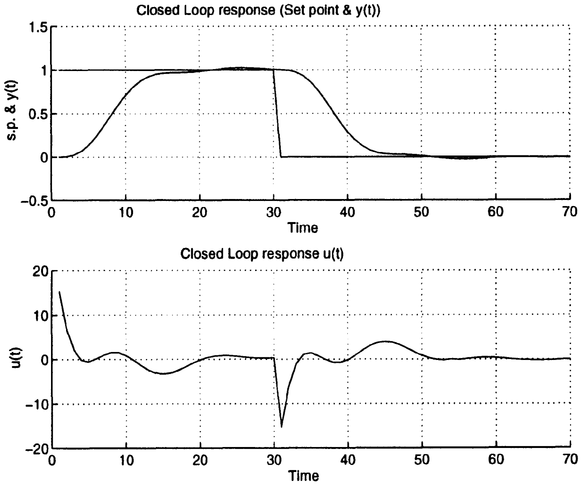


Fig. 6. GAGPC control of a non-linear process.

To decrease the prediction influence far into the future, α_i coefficients can be used in the cost index. If distant errors are given less weight, then some problems caused by model deficiencies can be avoided. These deficiencies are usually more appreciable when the time elapsed is longer.

Figure 7 shows an application example of this technique. Coefficient are set to $\alpha_j = 0.5^{j-1}$, the other parameters have values previously described. The results show, on the one hand, that output is a bit quicker (since the prediction horizon is attenuated), and on the other hand, future predictions are not so important. This aspect is interesting if the model of the system is not very reliable.

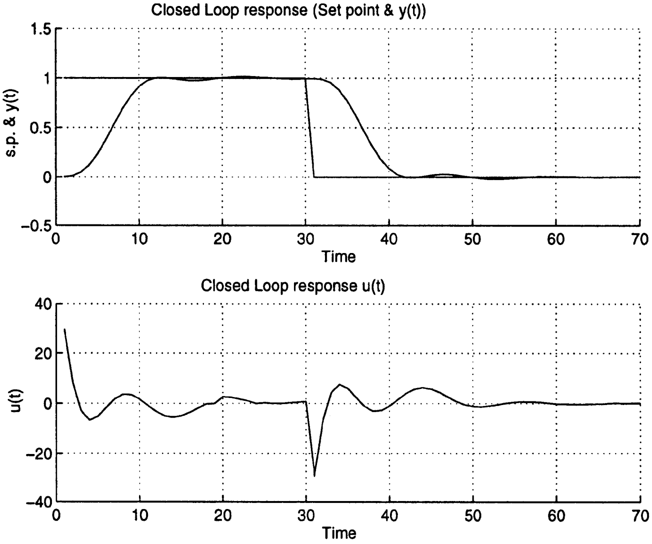


Fig. 7. GAGPC control of a non-linear process taking $\alpha_j = 0.5^{j-1}$.

5 Conclusions

The use of a numeric optimization technique such as Genetic Algorithms allows the use of any type of cost function. By applying this technique to the Generalized Predictive Control, a GAGPC is obtained. GAGPC has a more flexible cost function, and this means it can be applied in the predictive control of non-linear processes. Therefore, GAGPC offers more benefits when compared with other types of predictive controllers.

We know the GPC controller is quite robust. The GAGPC controller is as robust, and has the additional advantage that it can be applied to non-linear processes. This paper shows an example of a non-linear process controlled with a GAGPC. Despite the handicap of uncertainties or errors in the model, the closed loop maintained an appropriately dynamic behaviour. This feature makes GAGPC useful in most industrial control processes.

GAGPC has a disadvantage in some process control applications: the computational cost is noticeable. But this is compensated because GAGPC can give a control action at any time. Although this action is not optimal – it can be used to acceptably control a process in real-time.

References

1. F.X. Blasco. Controlador Predictivo Optimizado con Algoritmos Genéticos (GAGPC). Extensiones. Technical Report , Dept. Ingeniería de Sistemas Computadores y Automática. U. Politécnica de Valencia., 1995. (In spanish)
2. E.F. Camacho and C. Bourdons. *Model Predictive Control in the Process Industry.* Springer, 1995.
3. D. W. Clarke. Advances in Model-Based Predictive Control. In D. W. Clarke Editor, editor, *Advances in Model-Based Predictive Control*, pages 3–21. Oxford University Press, 1994.
4. D.E. Goldberg. *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley, 1989.
5. Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs.* Springer, 1996.
6. D.W. Clarke C. Mohtadi and P.S. Tuffs. Generalized Predictive Control-part I. *Automatica*, 23(2):137–148, 1987.
7. D.W. Clarke C. Mohtadi and P.S. Tuffs. Generalized Predictive Control-part II. Extensions and Interpretations. *Automatica*, 23(2):149–160, 1987.
8. K.R. Muske and J.B. Rawlings. Model Predictive Control with Linear Models. *AIChE Journal*, 39(2):262–287, 1993.
9. J.S. Senent. Controlador Predictivo Generalizado. Optimización mediante Algoritmos Genéticos (GAGPC). Technical Report , Dept. de Ingeniería de Sistemas Computadores y Automática, Universidad Politécnica de Valencia, 1995. (In spanish)
10. M. Martínez J.S. Senent and F.X. Blasco. A Comparative Study of Classical vs Genetic Algorithm Optimization applied in GPC Controller. *IFAC World congress*, 1996.

Piping Layout Wizard: Basic Concepts and Its Potential for Pipe Route Planning

Teruaki Ito
Department of Mechanical Engineering, The University of Tokushima
2-1, Minami-Josanjima, Tokushima 770-8506, Japan
Email: ito@me.tokushima-u.ac.jp

Abstract. The paper proposes a search method for pipe route planning using genetic algorithm incorporated with several heuristics. First, the basic principle of our method is presented using key ideas which include representation of pipe route for GA operations, spatial potential energy to cover design scenarios, fitness function, basic GA operations, coordinates conversion procedure, and route modification procedure using subgoal setting. In order to apply the method to actual problems and to solve them in a practical manner, the study employs various heuristics, which are concept of direction, generation of initial individuals using intermediate point, extended two-points crossover, and dynamic selection. Those heuristics are also described and their effectiveness in our method is discussed. Then, the paper shows a prototype system, or Piping Layout Wizard, which were developed based on our approach and discusses the validity of the proposed method.

1 Introduction

Layout designs, which effectively utilize the functionality of component equipments and appropriately satisfy the spatial constraints, is playing a key role in engineering designs of various production systems, chemical plants, power plants, factories, etc. For piping layout design [11] as one of these layout designs, a designer initially creates an appropriate design model and interactively modifies the model in a trial-and-error manner until the final completion of an approved design. However, keeping an appropriate space between the pipe and the wall or the obstacles under complicated spatial constraints, the optimal route path planning for the appropriate layout design has been a difficult task even to a skilled designer.

In the mean time, owing to the diversifying products and shortening life-cycle of the products, the conventional style of design using a series of consecutive procedures has become inadequate to satisfy those demands which require frequent changes of design specification even after the final design is settled. To comply with these circumstances, concurrent engineering [2], [10] has been proposed to use the idea to make decisions in the earlier phases of design. Consequently, conceptual design is becoming the more important. Piping layout design described in this paper is no exception. We believe that the conceptual design, or general understanding on the piping route path as a whole as well as technical understanding in details, plays a key role to comply smoothly and appropriately with design requirements.

The paper describes the method of piping route path planning using the optimization technique of genetic algorithm (GA) [1], [4], [5] as an approach to conceptual design on piping layout designs [6], [7], [8]. First, basic mechanism of piping route path generation using GA is described. Then, main features of our strategy applied to the basic mechanism is presented. The paper also presents the prototype system, or Piping Layout Wizard, which we are developing based on the methodology and shows the validity of the proposed approach.

2 Basic Mechanism Of Pipe Rroute Planning Using GA

2.1 Genetic Representation of Pipe Route

As an optimal search method for multiple peak functions, GA stemming from the generation of evolution of living things is applied to various optimization problems and its validity has been verified so far [3], [9], [12]. To apply GA to solve optimization problems in design, design parameters versus character sets of gene, or objective function versus fitness value must be considered and constraint conditions may be included in objective function under a penalty function. In this way, after repeating GA manipulations, a new character set which represents a new generation can reveal the appropriate design parameters.

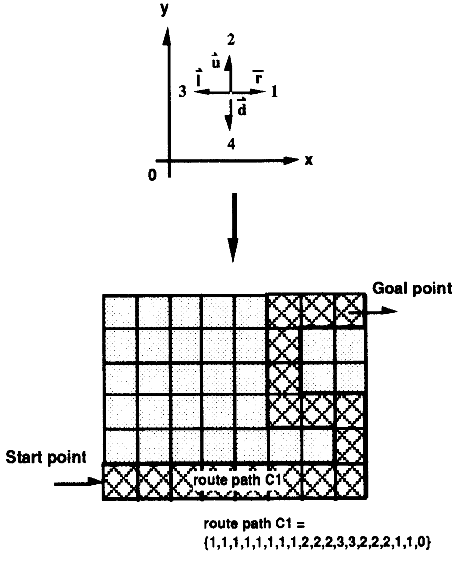


Figure 1 Genetic representation of pipe route

In order to use GA in piping route path planning as one of the optimization problems, a route path from a starting point to a destination point is represented by a character string and is regarded as a design parameter.

In our approach, a working space for piping route planning is represented by a 2D model and the space is divided into the cells of $m \times n$. A route path is represented using a combination of cells connecting a starting cell and a destination cell. To represent direction of route path, a unit vector set $\vec{v}=\{\vec{r},\vec{u},\vec{l},\vec{d},\vec{0}\}$ is defined, each vector represents right, up, left, down and stop, respectively, and character string of {1, 2, 3, 4, 0} corresponds to each vector. Using information on the cells which compose the route path, each individual is coded. For example, the gene type for a route path is expressed using symbols including {1, 2, 3, 4, 0},where zero means the current point already reached the destination cell. Figure 1 shows the vector set and an example of route path C_1 which is represented by C_1={1,1,1,...,2,1,0,0,0,0}.

2.2 Spatial Potential Energy in Pipe Route Generation

This section presents a brief overview of the concept of spatial potential energy which we adopted to make favorable route path. In route path planning, high priority is basically given to the shorter route path. In addition to this, various considerations are made; a route must go along the wall and obstacles as much close as possible, avoiding a diagonal path, etc. After these considerations, the most appropriate route path is designed. Using the concept of spatial potential, the degree of access to the wall or the obstacles is quantitatively calculated, and used as a part of objective function for the generation of piping route path using GA.

To determine the distribution of spatial potetial, the working space is divided into $m \times n$ (m=1,2,...; n=1,2,...) unit cells. For example, those cells which contain any portion of the obstacles are given the potential value P_n, and each cell is given the potential value of P_1, P_2,...,P_{n-1} according to the distance from the obstacle cell. Those cells which are located next to the wall are given the potential value P_0, which means that the route path is more favorable if it goes along with the wall. Only the positive values are used as a potential energy. The higher value means that the cell is far from the wall or the obstacles.

2.3 Studies on Crossover Method

Determination of crossover methods depends upon the application. For example, uni-crossover is suitable for the case that each gene has individual information, blend crossover is suitable for the case that genotype has a continuous value. In the case of piping route path, genotype shows a continuous value. A route path must connect a starting and a goal points. In addition to that, considering the layout of equipments, physical conditions for temperature, maintenance and so on, the most appropriate route path among various candidate route paths is designed. In other words, how to reach the goal point is the important thing to consider.

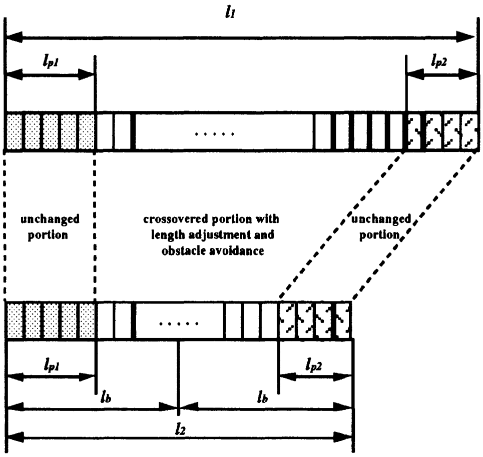


Figure 2 Crossover method for parents with different length

A route path must connect the two points or starting and goal. The length of genotype is variable and not fixed, which means that a genotype is elastic like a rubber band.

Sometimes, genes having different length must be crossed over. To cover the difference in length, we do as follows as shown in Figure 2. Set the length of parent 1 to $\mathbf{l}_1$, and that of parent 2 to $\mathbf{l}_2$ where $\mathbf{l}_1 \geq \mathbf{l}_2$. Then set the half of $\mathbf{l}_1$ to $\mathbf{l}_b$, multiply $\mathbf{l}_b$ with a random number between 0 and 1, and give $\mathbf{l}_p$. The portion between 0 and $\mathbf{l}_p$ and the portion between $\mathbf{l}_b$ and 1 are exchanged. Since the length of genes may differ each other, additional vector array may be inserted if the length of generated gene is too short to connect each end of the parent gene.

To avoid generating genes including obstacles, potential value of each cell is checked to see if the cell in on the obstacle. If it is, a vector is repeatedly generated until it does not on the obstacle. In this way, we could obtain those genes which does not contain obstacle cells in the earlier generations. As a result, a wide range of search area is considered in GA.

From the results of simulations, two-points crossover tries to avoid obstacles, aggressively find a new route path, and can find a better route path during an earlier generations. On the other hand, uni-crossover can inherit the characteristics from each parents at the same ratio, do not change the genotype, easily to conduct crossover operations even between parents with different lengths, and can generate various kinds of vector arrays. We mainly applied two-points crossover operation but also applied uni-crossover operation as a control.

2.4 Definition of Fitness Function

In general, piping route path planning is considered from various perspectives. Some of the typical perspectives to the planning would be (a) shorter length of route path, (b) arrangement of the pipes under the same categories, (c) guarantee the maintenance spaces, and so on. Our study considers the shorter length of route path and, from this respect, elements for fitness function were studied. We also considered that the route path should be as straight as possible but no diagonal path is permitted. The number of turning points in the path should be considered. The path should go along the surrounding wall in the work space or obstacles placed in the work space as close as possible. The potential energy is set lower in these cells. Considering these elements, fitness function was defined as shown in (2.1).

$$f(x) = f_0 + f_1 + p_{\max} + C + W \quad (2.1)$$

f_0 accumulates the number of cells and gives the length of route path. Length of route path is evaluated in f_0, but it is not enough to cover a variety of route paths. We applied the function f_1, which accumulates the total potential values for the route path and p_{max}, which considers the maximum value of potential energy in the cells of the route path. Every time the direction of route path is changed, a certain weight is added to the fitness value as C in (2.1). If a route path is on any obstacles, or the path contains cells including obstacles, a large weight is added to the fitness value as shown in (2.2).

$$W = p_{obstacle} \times A \quad (2.2)$$

3 Approach To Pipe Route Planning

The basic mechanism described in the previous section can give an idea to generate appropriate route paths connecting between the starting and goal points. In addition to this, we used our strategies to apply the idea to piping route path planning in order to generate paths more effectively. This sections describes some of the features in our strategies.

3.1 Introduction of Tendencies of Directions in Chromosome

A gene goes from a starting point towards a goal point. The most important thing is that a gene must reach the goal point for sure, which means that a gene possesses the characteristics that it moves towards the goal. Otherwise, a gene randomly goes inside the search area, which is time consuming and a waste of time.

We define the concept of "zone" to give chromosome the tendencies in the direction of route path from a current position towards the goal point. Using coordinates of current cell and goal cell, a zone is determined in each cell and priority vector is set to each zone. If the priority is set too high, however, all chromosomes have the same tendency in direction and route path cannot be appropriately generated. The appropriate priority was, therefore, set in a trial-and-error manner to generate chromosomes having variety of route paths.

Using the concept of tendencies in direction for route path, we could make route paths under control and give them the tendencies so that the paths are likely to go towards the goal point. Consequently, initial individuals are effectively generated. The method is also applied in our system when an additional portion is patched to cover the shortage of route path in crossover operations.

3.2 Introduction of Intermediate Point for Initial Individual Generation

Using a starting point and a goal point, initial individuals are randomly generated before GA procedure. First, the zone of a current cell is determined using coordinates of a starting and a goal points. A roulette based on the ratio of priority vector in the zone is set up, an arbitrary point on the roulette is determined using a random number generator between 0 and 1, and the first gene is selected. A current point is forwarded using the selected vector, and the coordinates of the updated current cell are obtained. Then the current zone is set up based on the updated current cell and the goal cell. The same procedure is repeated until the current point reaches the goal cell. In this way, initial individuals are generated.

In the mean time, to find the most appropriate route path, the initial route path should be as random as possible in the working area. Without considering obstacles, route paths are generated only by the specified cells for the starting and destination points. But most of the route paths in the initial individuals may have the tendency to go straight to the goal point. The route paths using these initial individuals did avoid obstacles and reached the goal point in the end. Judging from the route path length, the number of turnover and overall observations, the most appropriate route path is not always found.

To generate initial individuals more randomly, we defined intermediate points to be passed in the route path and applied in our approach. Making a bisector cutting through the line which connects the starting cell and the destination cell, an arbitrary cell on the bisector is selected. Making an arbitrary line passing through the bisector cell from the starting cell to the destination cell, initial individuals are generated referring to this line. Since the arbitrary cell is randomly selected, the route paths cover the overall working space.

3.3 Extended Crossover Methods

Using the appropriate initial individuals generated in the method mentioned in the previous section, we conducted simulation of piping route path planning using several crossover methods. As results, the followings were observed; (1) Uni-crossover converges individuals at the earlier generations; (2) Uni-crossover does not always exclude those individuals which include obstacle cells; (3) Two-points crossover is superior to uni-crossover.

Although two-points crossover generated appropriate route paths, some of them seem to be a locally optimized route paths. To avoid that, we applied dynamic selection ratio based on the minimum fitness value, average fitness value and the number of cells on obstacles. The first selection ratio of 40% is used until all of the individuals become obstacle free. Then the ratio is set down to 3% and to study all the possible route paths. When the convergence status becomes a certain level, the ratio is set back upto 40%. If the difference between the average fitness value and the minimum fitness value is below 5, we assumed that the convergence is going to terminate. In this way, we excluded those individuals including obstacles in the earlier generations, we tried to take time to find the most appropriate route path without converging to a locally optimized route path. When individuals are likely to converge to an appropriate route path, convergence speed is accelerated. We call our method as "extended crossover method" and applied to our prototype system.

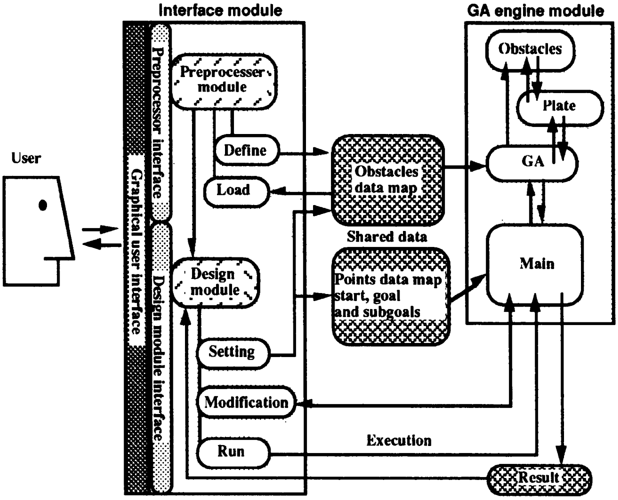
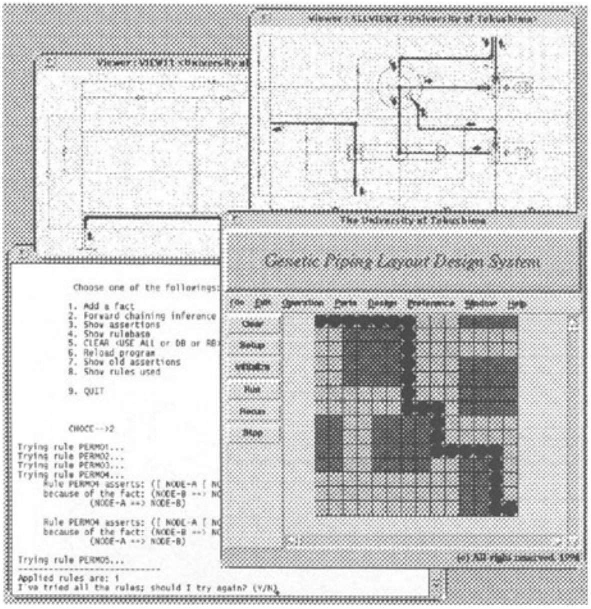


Figure 3 System overview of Piping Layout Wizard

4 Piping Route Path Planning System

We have developed a prototype system for piping route path planning using GA approach. The system is composed of two module layers as shown in Figure 3. The outer layer is a user interface module with which users directly interact using simple mouse operations and obtain the planning results displayed on the module. The inner layer is a GA engine module, which makes optimization of piping route path based on the parameters given by the users and returns the result back to the interface module.

As for the operation in using the system, a user determines the size of unit cell and divides the working area into **m**×**n** (m=1,2,...; n=1,2,...) using the unit cell. Considering the layout of machines or component equipments, the user arranges obstacles with mouse manipulation. Then, the user sets the starting and goal cells, the user may also set subgoal cells if necessary, and stores the parameters. The optimized route paths generated only based on the starting and goal points are not always appropriate in terms of piping route path planning. For example, even if a path is not the shortest of all the candidates route paths, it might be an appropriate path because it


The University of Tokushima
Genetic Piping Layout Design System
File Edit Operation Parts Design Preference Window Help
Clear
Setup
Run
Stop

```
          Choose one of the followings:

          1. Add a fact
          2. Forward chaining inference
          3. Show assertions
          4. Show rulebase
          5. CLEAR <USE ALL or DB or RB
          6. Reload program
          7. Show old assertions
          8. Show rules used

          9. QUIT



          CHOCE-->2

Trying rule PERM01...
Trying rule PERM02...
Trying rule PERM03...
Trying rule PERM04...
     Rule PERM04 asserts: ([ NODE-A [ N
     because of the fact: (NODE-B --> N
               (NODE-A --> NODE-B)

     Rule PERM04 asserts: ([ NODE-A [ N
     because of the fact: (NODE-B --> N
               (NODE-A --> NODE-B)

Trying rule PERM05...
------------------------------
Applied rules are: 1
I've tried all the rules; should I try again? (Y/N)
```

Figure 4 Snapshot of pipe route planning operation

goes through a certain point. We have realized the setting of subgoal in GA route path planning.

The GA engine will generate route paths based on the given parameters and display candidate paths for the route planning. Figure 4 shows a snapshot of pipe route planning operation. Figure 4 also shows viewer module and inference module which are under development to support GA module, but those modules are beyond the scope of this paper and will be discussed elsewhere.

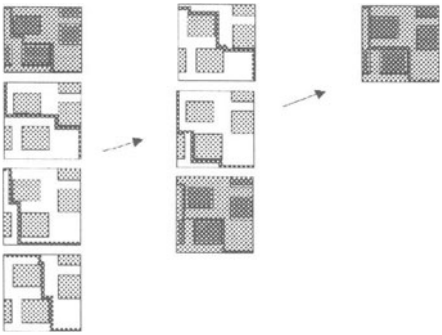


Figure 5 Candidate routes during convergence with GA operation

The paths proposed by the system include not only the finally optimized route path but also those paths which were generated during GA operations before the convergence of route paths. It is the designer who evaluates those paths and decides to adopt one of these paths, or make another try to consider it from some different perspectives. Figure 5 shows an example of route path planning using the system. Several candidate route paths are displayed during the optimization procedures.

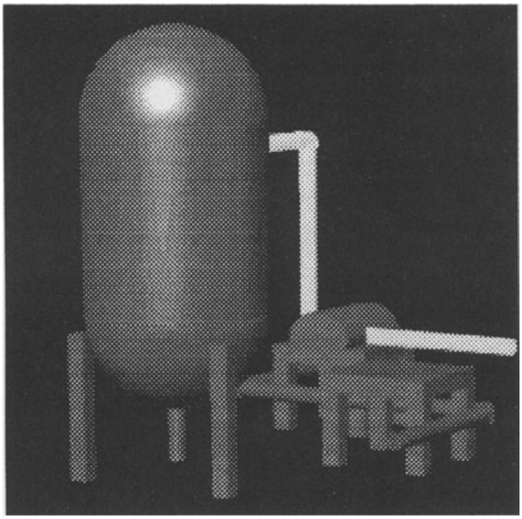


Figure 6 An example of route planning under integrated environment

Integrating the systems with other design tools such as 3D CAD or a CG tool, route path planning using our system will provide more visualized environments where the users are stimulated their ideas in designing the appropriate route path. Figure 6 shows an example of route path planning using the integrated environments.

5 Concluding Remarks

The paper proposed a search method for pipe route planning using genetic algorithm incorporated with several heuristics. The basic principle of our method was presented using key ideas which include representation of pipe route for GA operations, spatial potential energy to cover design scenarios, fitness function, basic GA operations, coordinates conversion procedure, and route modification procedure using subgoal setting. The paper also describes various heuristics of our method, which are concept of direction, generation of initial individuals using intermediate point, extended two-points crossover, and dynamic selection. We have developed a prototype system, or Piping Layout Wizard, based on our approach and presented the validity of the proposed method.

For the future works, we will study higher performance of processes to cope with interactive designing, consideration of more complicated conditions, route path planning in 3D spaces, and integration with 3D CAD or CG tools.

References

1. Agui, T. and Nagao, T.: Genetic Algorithm. Konando (1993) (in Japanese)
2. Fukuda, S.: Concurrent Engineering. Baifukan (1993) (in Japanese)
3. Fujita, K., Akagi, S. and Hirokawa, K.: Approach of Optimal Nesting Using Genetic Algorithm and Local Minimization Algorithm. Transactions of JSME, Vol.59, No.564(C): (1993) 2576-2583 (in Japanese)
4. Goldberg, D.E.: Genetic Algorithm in Search, Optimization and Machine Learning. Addison-Wesley (1989)
5. Goldberg, D.E.: Genetic and Evolutionary Algorithms Come of Age. Communications of the ACM, Vol.37, No.3. (1994) 113-119
6. Ito, T. and Fukuda, S.: Hybrid Approach to Piping Route Path Design Using GA-based Inspiration and Rule-based Inference. Advances in Concurrent Engineering - CE97, Technomic Publishing. (1997) 237-244
7. Ito, T. and Fukuda, S.: Piping Route Path Planning using Genetic Algorithm. Proceedings of Design Engineering Technical Conferences, Sep.14-17, Sacramento, U.S.A. (1997) EIM3725
8. Ito, T. and Fukuda, S.: A Study on Search Method for Pipe Route Planning Using Genetic Algorithm. Journal of Japanese Society for Artificial Intelligence, Vol.13, No.5 (1998) (in printing) (in Japanese)
9. Matsumura, T. and Usui, E.: On the Development of Autonomous Operation Planning System with Genetic Algorithm. Journal of the Japan Society of Precision Engineering, Vol.62, No.7. (1996) 974-978 (in Japanese)
10. Prasad, B.: Concurrent Engineering Fundamentals. Prentice Hall (1996)
11. Takakuwa, T.: Analysis and Design for Drainpipe Networks. Morikita Shoten (1978) (in Japanese)
12. Yamamura, M. and Kobayashi, S.: Toward Application Methodology of Genetic Algorithms. Journal of Japanese Society for Artificial Intelligence, Vol.9, No.4. (1994) 506-511 (in Japanese)

Integration of Constraint Programming and Evolution Programs: Application to Channel Routing

Alvaro Ruiz-Andino[1] and José J. Ruz[2]

[1] Department of Computer Science
[2] Department of Computer Architecture
Universidad Complutense de Madrid
arai, jjruz@eucmax.sim.ucm.es. http://mozart.mat.ucm.es/ alvaro

Abstract. This paper presents a novel approach for the integration of constraint programming techniques and evolution programs. In this approach the genetic operators implementation is based on a constraint solver, and chromosomes are arc-consistent solutions to the problem, represented as arrays of finite integer domains. This method allows to tackle efficiently constrained optimisation problems over finite integer domains with a large search space. The paper describes the main issues arising in this integration: chromosome representation and evaluation, selection and replacement strategies, and genetic operators design. The implemented system has been applied to the channel routing problem, a particular kind of the interconnection routing problem, one of the major tasks in the physical design of very large scale integration circuits.

1 Introduction

Evolutionary computation techniques are global optimisation methods for complex objective functions over very large search spaces. They come, as held in [11], from the refinement of genetic algorithms. An evolution program maintains a population of *chromosomes* $\mathcal{P}(t) \equiv \{X_1, \cdots, X_n\}$ for each iteration t. Each chromosome represents a potential solution to the problem at hand, implemented as a complex data structure $\mathcal{A}$. The initial population $\mathcal{P}(0)$ is generated randomly or by any other method. The population is evaluated, computing a *fitness* value for each chromosome X_i which indicates the goodness of the chromosome as a solution to the optimisation problem. Then a new population, $\mathcal{P}(t+1)$, is formed by selecting some chromosomes from $\mathcal{P}(t)$. Best fitted solutions are more likely to be selected. Some members of the population undergo transformations by means of *genetic operators* to form new chromosomes that take part in the next population. There are binary genetic operators (crossover type), which create a new solution by combining two selected chromosomes, and unary operators (mutation type), which create a new solutions by a small change in a single chromosome. After some number of generations (iterations) the program converges – it is hoped that the best chromosome represents a near-optimum solution. There is a theoretical foundation for this kind of algorithms based in the schemata theorem, which is beyond the scope of this introduction.

Handling of constraints in evolution programs introduces an additional complexity in the design of the genetic operators. In constrained problems, a minimal change to a feasible solution is very likely to generate an unfeasible one, but unfeasible solutions cannot simply be dropped from the solution space because doing so would prevent certain good solutions from being generated. During the past few years, several methods have been proposed for handling constraints by genetic algorithms for optimisation problems. These methods can be grouped in the following three categories: 1) Specialised operators that transform feasible individuals into new feasible individuals [11], 2) Penalty functions that reduce the fitness value of unfeasible solutions [8,6], and 3) Repairing unfeasible individuals [13,12]. However, all these approaches require problem specific programming. In this paper, we propose a general method for handling constraints in genetic algorithms and its application to channel routing [2], a NP-complete problem that can be stated as a constrained optimisation problem over finite integer domains. The method involves the integration of a constraint solver over finite integer domains into evolution programs.

The integration of constraint solving and genetic algorithms was first proposed by Paredis in [15] where a genetic algorithm is enhanced with forward checking to create partially instatiated chromosomes with unknown valued genes. However, our method implements a general constraint solver based on the *indexical scheme*, used both for constraint propagation and constraint entailment, and it employs domain-valued genes.

The rest of the paper is organised as follows. Section 2 introduces constraint programming and the constraint solver used in this work. Section 3 describes in detail the integration of a constraint solver with an evolution program. Sections 4 explains the channel routing application and the results. Finally Section 5 presents the conclusions.

2 Constraint Programming

Constraint Programming over finite domains (CP(FD)) has been used for specifying and solving complex constrained optimisation problems as resource allocation, scheduling and hardware design [19]. Thanks to their combination of constraint solving and declarative style, CP languages are able to find solutions to combinatorial problems comparable in efficiency to procedural languages, yet requiring a much shorter development time. Most CP(FD) languages are based on a constraint propagation solver, an efficient and general approach to solve finite domain constraint satisfaction problems (CSP). Propagation-based solvers eliminate inconsistent values from the solution space, reducing the size of the search space both before and while searching.

A constraint satisfaction problem over finite integer domains may be stated as follows. Given a tuple $\langle \mathcal{V}, \mathcal{D}, \mathcal{C} \rangle$, where $\mathcal{V} \equiv \{v_1, \cdots, v_n\}$, is a set of domain variables; $\mathcal{D} \equiv \{d_1, \cdots, d_n\}$, is the set of an initial *finite integer domain* (finite set of integers) for each variable; and $\mathcal{C} \equiv \{c_1, \cdots, c_m\}$, is a set of constraints among the variables in $\mathcal{V}$. A constraint $c \equiv (V_c, R_c)$ is defined by a subset of variables $V_c \subseteq \mathcal{V}$, and a subset of allowed tuples of values $R_c \subseteq \bigotimes_{i \in \{j/v_j \in V_c\}} d_i$, where $\bigotimes$ denotes Cartesian product. The goal is to find an assignment for each variable $v_i \in \mathcal{V}$ of a value from each $d_i \in \mathcal{D}$ which satisfies every constraint $c_i \in \mathcal{C}$.

The basis for constraint solvers, as [7,4], is an arc-consistency algorithm [14, 17] that eliminates inconsistent values from variables' domains. A constraint $c \in \mathcal{C}$ relating variables v_i and $v_j \in \mathcal{V}$, is arc-consistent with respect to domains d_i, d_j iff for all $a \in d_i$ there exists $b \in d_j$ such that (a, b) satisfies the constraint c, and for all $b \in d_j$ there exists $a \in d_i$ such that (a, b) satisfies the constraint c. A CSP is called arc-consistent iff all $c_i \in \mathcal{C}$ are arc-consistent with respect to $\mathcal{D}$. Our starting point is constraint propagation solver which implements arc-consistency using the *indexical scheme* [18]. In this scheme, a constraint is translated into a set of reactive functional expressions, called *indexicals*, which maintain arc-consistency. An indexical has the form 'v in $E(V)$', where $v \in \mathcal{V}$, $V \subseteq \mathcal{V}$, and $E(V)$ is a monotonic functional expression which returns a finite integer domain. The solver is extended with constraint entailment detection, as presented in [3]. Blocking implication, constraint reiffication, constructive disjunction, and other powerful constraint handling techniques are built on entailment checking.

A *store* $\mathcal{S}$ is an array of finite integer domains, one for each variable. The execution of an indexical is triggered by changes in the domains of its set of arguments V in a data driven way. Indexicals can play one of two roles: propagating indexicals for arc-consistency maintenance, and checking indexicals for entailment checking. When a propagating indexical is executed $v_{\mathcal{S}}$ (the domain of v in the store $\mathcal{S}$) is updated with $v_{\mathcal{S}} \cap E_{\mathcal{S}}$, where $E_{\mathcal{S}}$ denotes the evaluation of $E(V)$ with the current variables' domains in $\mathcal{S}$. When a checking indexical is executed, if the domain of v is contained in $E_{\mathcal{S}}$, the constraint corresponding to the indexical is detected as entailed.

Figure 1 shows a simplified constraint propagation algorithm. It takes as input arguments a set of constraints to be satisfied (expressed as indexicals) and an initial store (an initial finite integer domain for each variable). It returns a new store where the domain for each variable has been pruned in such a way that arc-consistency is achieved, or NULL if inconsistency is detected (a variable was pruned to an empty domain). A propagation queue is used to schedule the execution of indexicals. Whenever a variable domain v_i is modified, all indexicals 'v_j in $E(V)$', such that $v_i \in V$, are queued.

In order to tackle constraint optimisations problems, which require not just a feasible solution but an optimal one according to some objective function, most constraint systems incorporate some kind of branch and bound algorithm. However, in real scale problems, the search space can be too large to use this kind of exhaustive bounded search. Our constraint system incorporates the arc-consistent evolution program optimisation presented in this paper.

```
function Arc-Consistency( $InitialStore$, $ConstrSet$ ) : Store;
begin
   Store := Init_Store($InitialStore$);
   Reset(Queue);
   for each $Ind_i \in ConstrSet$ do Insert($Ind_i$, Queue); end-for;
   while Not Empty(Queue) do
      Pop(Queue, '$v$ in $E(V)$');
      NewDomain := Store[$v$] $\cap$ Eval($E(V)$, Store);
      if Empty(NewDomain) return NULL; end-if;
      if (NewDomain $\subset$ Store[$v$]) then
         Store[$v$] := NewDomain;
         for each $Ind_i \in ConstrSet$ which depends on $v$ do
            Insert($Ind_i$, Queue);
         end-for;
      end-if;
   end-while;
   return Store;
end;
```

Fig. 1. Constraint propagation solver

3 Integration of CSP and Evolution Programs

Constraint propagation techniques and evolution programs complement each other. Constraint solving techniques opens an flexible and efficient way to handle constraints in evolution programs, while evolution programs allow searching for solutions in large scale constrained optimisation problems. This integration implies coming up with a solution for chromosome representation, chromosome evaluation, and genetic operators' design.

3.1 Chromosome Representation

The first point in the design of an evolution program is to define the data structure $\mathcal{A}$ to represent chromosomes. As we are dealing with integer valued variables, the naive representation would be an array of integers, one for each variable. But this representation would prevent taking advantage of the arc-consistency techniques embedded in the constraint solver. A chromosome will be implemented as an store, that is, an array of finite integer domains, rather than an array of integer values. Moreover, all chromosomes in the population will be arc-consistent solutions (AC-solutions) to the problem, generated by means of genetic operators based on the `Arc-Consistency()` function, which returns stores where inconsistent values have been removed from variables' domains. Therefore, a chromosome will be a sub-space of the search space containing many or none solutions, and in particular, it may be a fully determined (thus feasible) solution when all domains are singleton (a single integer). Evaluation, selection and replacement strategies assure convergence to fully determined solutions. Given this chromosome representation, genetic operators are arc-consistent search procedures that take as input argument an AC-solution, and search for a new AC-solution.

3.2 Chromosome Evaluation

In a constrained optimisation problem over finite integer domains the objective function may be assimilated to a domain variable v_{of} without loss of generality. Each AC-solution has a related domain d_{of} which is the pruned domain of the variable to optimise. The population is evaluated every generation, computing a fitness value for each chromosome. Fitness indicates how good a chromosome is as a potential solution to the problem. Hence, it is computed from the value of d_{of} in the AC-solution being evaluated, normalised with respect to the initial value of d_{of}. But as we are dealing with non fully determined solutions, it is necessary to measure and to take into account how 'realistic' the fitness value is, that is, the probability that a feasible solution lies within the AC-solution (a search sub-space), and how far we are from reaching it. This is why we use a dual chromosome evaluation, introducing a so-called *feasible* value as an approximate measure for this purpose. It is computed as normalised value denoting how much the variables' domains have been pruned. Chromosomes closer to be determined have a higher feasible value. Figure 2 shows the algorithms for computing fitness and feasible values.

```
function AC-fitness( AC-Solution, IniStore, Obj_Var ) : real;
begin
  MinO := min(IniStore[Obj_Var]); MaxO := max(IniStore[Obj_Var]);
  Min := min(AC-Solution[Obj_Var]); Max := max(AC-Solution[Obj_Var]);
  return (Min + $\lambda$*(Max-Min) - MinO) / (MaxO-MinO);
end;

function AC-feasible ( AC-Solution, IniStore ) : real;
begin
  return 1 - $\sum_i$((size(AC-Solution[$v_i$])-1) / (size(IniStore[$v_i$])-1));
end;
```

Fig. 2. AC evaluation: fitness and feasible

3.3 Integration Algorithm

Figure 3 shows the main algorithm of the Arc-Consistent Evolution Program. It follows the structure of a steady-state genetic algorithm. Functions in bold face invoke the propagation solver. Words in italics are parameters to be set.

The *initialise* step is a loop that generates a certain number (*population_size*) of chromosomes by means of the `AC-random-search()` procedure. *evaluate* step computes the AC-fitness() and AC-feasible() values for each AC-Solution (chromosome) in the population.

```
function AC-evolution-program ( InitialStore, ConstrSet,
                                Objective_Variable ) : Solution;
Var   $X1$, $X2$ : AC-Solution;
      $\mathcal{P}$ : Array[1..population_size] of AC-Solution;
begin
  initialise $\mathcal{P}$: for i := 1 to population_size do
    $\mathcal{P}$[i] := AC-random-search(InitialStore, ConstrSet);
  end-for;
  evaluate $\mathcal{P}$: for i := 1 to population_size do
    Fitness[i] := AC-fitness($\mathcal{P}$[i],InitialStore,Objective_Variable);
    Feasible[i] := AC-feasible($\mathcal{P}$[i],InitialStore);
  end-for;
  main loop: for t := 0 to max_iter do
    select: for i := 1 to survival_rate * population_size do
      X1 := biased-selection($\mathcal{P}$,fit_weight(t)*Fitness+Feasible);
      mark X1 to survive;
    end-for;
    mark best singleton chromosome in $\mathcal{P}$ to survive;
    alter $\mathcal{P}$: for i := 1 to population_size do
      if not marked-to-survive($\mathcal{P}$[i]) then
        X1 := biased-selection($\mathcal{P}$, Fitness);
        X2 := biased-selection($\mathcal{P}$, Feasible);
        $\mathcal{P}$[i] will-be-replaced-by AC-crossover(X1,X2,ConstrSet);
      else
        if random-real-between(0,1) $>$ mutation_rate then
          $\mathcal{P}$[i] will-be-replaced-by
            AC-mutation($\mathcal{P}$[i],ConstrSet,Fitness[i],Feasible[i]);
        end-if;
      end-if;
    end-for;
    update $\mathcal{P}$; evaluate $\mathcal{P}$;
  end-for;
  return best singleton chromosome in $\mathcal{P}$;
end;
```

Fig. 3. AC evolution program

In the *select* step, some chromosomes are marked as survivors, so they won't be replaced by the new AC-solutions generated by crossover. Best chromosomes are more likely to be selected, as biased-selection() procedure performs a biased stochastic selection. An elitist policy is used, so the best chromosome always survive. The *alter* step is divided in the two genetic operators, AC-crossover() and AC-mutation(), both AC-solution generators. New chromosomes generated by crossover take the place in the population of those chromosomes not chosen to survive. New AC-solutions generated by means of the mutation operators replace the chromosome used to generate it.

3.4 Selection and Replacement Strategies

In two points of the evolution program, a particular chromosome from the current population has to be selected: to survive, and for crossing-over. It will be a random selection but biased according to a certain criteria. Chromosomes with a higher criteria value are more likely to be chosen, but no good chromosomes may also be selected. Bias criteria for chromosomes to survive is a linear combination of fitness and feasibility. As the number of generations increases the weight of fitness (fit_weight) is increased to help convergence towards determined solutions. For each crossover two parents are selected, one biased on a fitness criteria and the other one biased on a feasibility criteria. Function `biased-selection()` is implemented as a quadratic biased rank selection.

3.5 Initial population generation

The first step in all evolution optimisation strategies is obtaining an initial population. Each chromosome of the initial population is generated by means of the `AC-random-search()`. This heuristic stochastic search procedure generates a new AC-solution which is a randomly generated sub-space of the AC-solution taken as input argument. To generate the initial members of the population, `AC-random-search()` is called with the whole search space as input argument. Figure 4 shows `AC-random-search()` algorithm. In first place, the input AC-solution is posted, and then a solution is searched for. A straight-forward way to implement an AC-solution generator is a variable-value choice heuristic. Variable ordering is randomly established. Domain to be sequentially assigned to each variable $v_{perm[i]}$ is a singleton value randomly chosen from variable's current domain. If the assignment leads to inconsistency, variable's domain is left unchanged, hence backtracking never occurs.

3.6 Genetic operators

Genetic operators generate the new chromosomes that will be added to population $\mathcal{P}(t)$ from chromosomes from population $\mathcal{P}(t-1)$. Genetic operators search for solutions whose arc-consistency is guaranteed by the propagation solver. Genetic operators implement stochastic heuristic searches, taking previous AC-solutions as an input information to guide the search for a new AC-solution. Genetic operators are clustered in two classes: mutation and crossover. Mutation operator creates a new AC-solution by means of a small change in a single chromosome, whereas crossover operator searches for a new AC-solution combining information from two chromosomes. The design of genetic operators is a crucial point in all evolution algorithms because they must guarantee both, new individuals to inherit their ancestors' properties, and to allow the exploration of new areas of the search space.

```
function AC-random-search(AC-Solution,ConstrSet): AC-solution;
begin
   NewStore := Arc-Consistency(AC-Solution, ConstrSet);
   perm[1..n] := random-permutation-between(1, n);
   for i := 1 to n do
      K := random-int-between(min(NewStore[$v_{perm[i]}$]),
                              max(NewStore[$v_{perm[i]}$]));
      Store := Arc-Consistency(NewStore, ConstrSet $\cup$ $\{v_{perm[i]}$ := K$\}$);
      if Store <> NULL then NewStore := Store; end-if;
   end-for;
   return NewStore;
end;
```

Fig. 4. AC random search

Crossover. New chromosomes generated by means of the crossover operator replace those not selected to survive. As shown in Figure 3, parent chromosomes are chosen using the `biased-selection()` function. One parent is selected biased towards fitness (probably a high fitted but not feasible solution) and the other parent is selected biased towards feasibility (probably a feasible solution, but with a low fitness value).

```
function AC-crossover( AC-Sol1, AC-Sol2, ConstrSet ) : AC-solution;
begin
   perm[1..n] = random-permutation-between(1, n);
   K := random-int-between(1, n);
   for i = 1 to K do
      ConstrSet := ConstrSet $\cup$ $\{v_{perm[i]}$ in AC-Sol1[perm[i]]$\}$;
   end-for;
   NewStore := Arc-Consistency( NULL, ConstrSet );
   for i = K+1 to n do
      ConstrSet := ConstrSet $\cup$ $\{v_{perm[i]}$ in AC-Sol2[perm[i]]$\}$;
      Store := Arc-Consistency( NewStore, ConstrSet );
      if Store <> NULL then NewStore := Store; end-if;
   end-for;
   return NewStore;
end;
```

Fig. 5. AC-Crossover

Different crossovers operators were tried on a set of benchmarks; an uniform AC-crossover search algorithm, as implemented in Figure 5, showed best average results. Given two AC-solutions, the AC-crossover operator generates a new AC-solution, which is a mixture of the two parents. K (a random value between 1 algorithms have been proposed for the problem [20, 5], most of them graph-based. More recently neural network [16], simulated annealing [1], and genetic algorithms [10], have been proposed.

The above formulation can be stated as a constrained optimisation problem over finite domains. Each net is to be assigned a pair layer/track where the horizontal segment is to be placed. Let L be the number of layers and let T be the number of tracks in each layer. Each net is associated a domain variable v whose initial domain ranges from 0 to $L \times T - 1$, indicating the global position of the track where the horizontal segment is to be placed. The layer is $v_i//T$, where // stand for integer division, and the track number within the layer is v_i *mod* T, where *mod* stands for the remainder operation.

Given this problem representation, the set of domain variables $\mathcal{V}$ must satisfy the following two sets of constraints: 1) horizontal constraints, to avoid overlapping of horizontal segments, and 2) vertical constraints to avoid overlapping vertical segments. Horizontal constraints are straight-forward: for each pair of nets n_i and n_j, if the rightmost terminal of n_i is equal or greater than the leftmost terminal of n_j, nets n_i and n_j cannot be assigned the same global track, that is, $v_i \neq v_j$. Vertical constraints require a conditional precedence relation of tracks: for each terminal i, let net_{top} be the net where top terminal i belongs to, and let net_{bot} be the net where bottom terminal i belongs to. If both nets are placed in the same layer, the track assigned to net_{top} must be grater than the track assigned to net_{bot}.

4.2 Benchmark

The benchmark suite given in [20] is well used in the VLSI design community, among which the Deutch's difficult problem is a representative one. The problem is to route a set of 72 nets on a channel where there are 174 terminals on each row. There are 117 vertical constraints and 846 horizontal constraints. The objective function to be minimised is the sum of the lengths of each routing path for a given number of tracks per layer.

The problem was tested for 2 layers/11 tracks, 3 layers/7 tracks, and 4 layers/5 tracks. For each number of layers the program was run ten times with a different initial random seed on a SPARC-10 workstation, one hour of CPU time each run. *population_size* (600), *survival_rate* (0.5), and *max_iter* (125) parameters were set in order to meet CPU time limit. *mutation_rate* was set to 0.05, and *fit_weight* ranged from 0.4 (initial) to 0.8 (final).

We have also implemented a parallel version of the program on a CRAY T3E massively parallel processor. Iterations of the alter loop (execution of genetic operators) are executed in parallel. Execution time with 4 processors (DEC Alpha) falls down to 10 minutes.

Table 1 reports the length of the best (CP+EP best) and worst (CP+EP worst) solutions found for each number of layers, and compares them with best results reported by [21] for one hour SPARC-10 CPU time, using a constraint programming approach with a Branch&Bound algorithm (CP+B&B (Zhou)). Figure 6 show the best solutions found for 3 and 4 layers.

and n) randomly chosen variables $v_{perm[1]}$ through $v_{perm[K]}$ are constrained to domains from first parent. Remaining variables $v_{perm[K+1]}$ through $v_{perm[n]}$ are constrained, if possible, to the corresponding second parent's set of domains. If constraint posting leads to inconsistency, variable's domain is left unchanged, hence backtracking never occurs.

Mutation is applied, with a probability *mutation_rate*, to those chromosomes not selected to survive. Different mutation heuristics are used depending on quality of AC-solution to be replaced. A random local search is performed around fitted and feasible AC-solutions. Determined but unfitted AC-solutions are transformed producing a not determined AC-solution, making possible the exploration of new areas of the search space. Finally, bad and unfeasible AC-solutions are replaced by new AC-solutions generated by `AC-random-search()`.

4 Channel Routing Application

The integration presented have been applied to channel routing, a particular kind of the interconnection routing problem, which is one of the major tasks in the physical design of very large scale integration (VLSI) circuits.

4.1 Problem Description

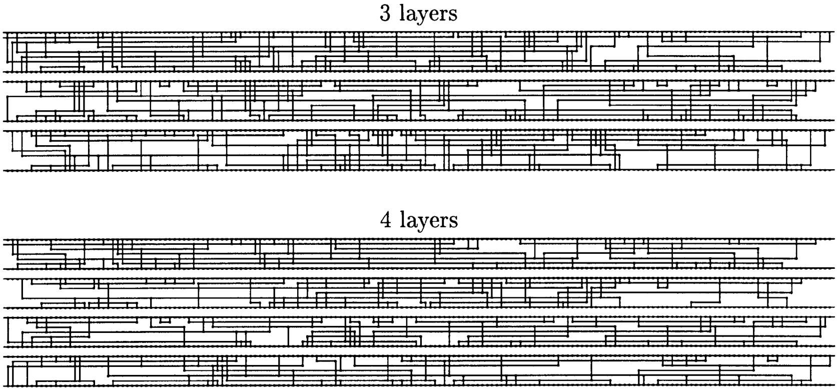
Channel routing problem restricts the routing area to a rectangular channel. A channel consists of two parallel horizontal rows with numbered pins. Pins that belong to the same net are connected together subject to a set of routing constraints. The channel routing problem is to find routing paths for a given set of nets in a given channel such that no segments overlap each other, and the routing area and the total length of routing paths are minimised. There are different approaches to the problem that impose different restrictions on the channel and routing paths. In this paper we consider the dogleg-free multilayer channel routing problem which impose the following three restrictions: First, the routing area in a channel is divided into several pairs of layers, one called a horizontal layer and the other a vertical layer. There are several tracks in each horizontal layer. Minimising the routing area means minimising the number of tracks. Second, the routing path for every net consists of only one horizontal segment which is parallel to the two rows of the channel, and several vertical segments which are perpendicular to the two rows. Horizontal segments are placed only in horizontal layers and vertical segments are placed only in vertical layers. The ends of segments in a routing path are connected through via holes. Third, no routing path can stretch over more than one pair of layers. Thus, for each net, we only need to determine the horizontal layer and the track for the horizontal segment. The positions for the vertical segments are determined directly after the horizontal segment is fixed.

The channel routing problem has been studied extensively in the VLSI design community. LaPaugh [9] has proved that the problem is NP complete. Many

	2 Layers	3 Layers	4 Layers
CP+EP best	3923	3523	3311
CP+EP worst	3935	3531	3329
CP+B&B (Zhou)	4102	3567	3364

Table 1. Experimental results comparation

The embodiment of evolution optimisation in constraint programming obtains better results than Branch&Bound (B&B) optimisation. Moreover, B&B would become an ineffective approach for larger scale problems.


3 layers

4 layers

Fig. 6. Best routings found

5 Conclusions

We have described a method for solving constrained optimisation search problems based on the integration of constraint programming and evolution programs. On one hand, the integration permits to tackle, in a constraint programming environment, optimisation problems whose size is inaccessible to exhaustive enumeration techniques like Branch&Bound. On the other hand, it provides a general and effective method to deal with constraints in evolution programs. This integration have been implemented with our own constraint solver, but it could be ported to any commercial constraint programming environment. The integration method have been applied to channel routing, a particular kind of the interconnection routing problem, which is one of the major tasks in the physical design of very large scale integration (VLSI) circuits. Constraint programming allows a declarative problem statement, flexible enough to be adapted to new technological constraints, and evolution programs are a powerful optimisation technique for large search spaces. We believe this approach promise to be a useful tool in VLSI design.

References

1. Brouwer, R.J., Banerjee, P.: Simulated annealing algorithm for channel routing on a hypercube multiprocessor. IEEE Int. Conf. on Computing Design. (1988) 4-7.
2. Burnstein, M.: Channel routing. Layout design and verification. North-Holland (1986) 133-167.
3. Carlson, B.: Compiling and Executing Finite Domain Constraints. PhD Thesis, Computer Science Department, Uppsala Unv. (1995).
4. Dincbas,M., Van Henteryck,P., Simmons,H., Aggoun,A.: The Constraint Programming Language CHIP. Proc. of the 2nd Int. Conf. on 5th Generation Computer Systems. (1988), 249-264.
5. Fang,S.C., Feng,W.S., Lee,S.L.: A new efficient approach to multilayer channel routing problem. 29th ACM-IEEE Design Automation Conf. (1992), 579-584.
6. Homaifar, A., Lai, S.H., Qi, X.: Constrained optimisation by genetic algorithms. Simulation, 62(4) (1994) 242-254.
7. ILOG. ILOG Solver C++. Reference Manual, ILOG S.A. (1993).
8. Joines, J., Houck C.: On the use of non-stationary penalty functions to solve nonlinear constrained optimisation problems with gas. Proc. of the 1st IEEE Int. Conf. on Evolutionary Computation. Piscatway N.Y. IEEE Press (1994) 579-584.
9. LaPaugh, A.S.: Algorithms for integrated circuits layouts: an analytical approach. PhD Dissertation, MIT Lab. of Computer Science (1980).
10. Liu, X., Sakamoto, A., Shimamoto, T.: Genetic channel routing. IEICE Trans. Fundamentals (1994) 492-501.
11. Michalewicz, Z.: Genetic algorithms + Data Structures = Evolution Programs. 2nd Edition, Springer-Verlag (1994).
12. Michalewicz, Z., Nazhiyath, G.: Genocop III. A co-evolutionary algorithm for numerial optimisation problems with non-linear constraints. Proc. of the 2nd IEEE Int. Conf. on Evolutionary Computation. NY. IEEE Press (1995) 647-651.
13. Minton, D., Johnston, M.D., Philips, A.B., Laird, P.: Minimising conflicts: a Heuristic Repair Method for constraint satisfaction and scheduling problems. Artificial Intelligence 58 (1992).
14. Mohr, R., Henderson, T.C.: Arc and path consistency revisited. Artificial Intelligence 28 (1996) 225-233.
15. Paredis, J.: Genetic State-Search for constrained Optimisation Problems. 13th Int. Joint Conf. on Artificial Intelligence (1993).
16. Takefuji: Neural network parallel computing. Kluwer Academic Press (1992).
17. Van Hentenryck P., Deville, Y., Teng C.M.: A generic Arc-consistency Algorithm and its Specialisations. Artificial Intelligence 57 (1992) 291-321.
18. Van Hentenryck, P, Saraswat, V., Deville, Y.: Design, Implementation and Evaluation of the Constraint Language cc(FD). Draft (1993).
19. Wallace, M.: Constraints in Planing, Scheduling and Placement Problems. Constraint Programming, Springer-Verlag (1994).
20. Yoshimura, T., Kuh, E.S.: Efficient algorithms for channel routing. IEEE Trans. on CAD. (1982) 25-35.
21. Zhou, N.F.: Channel Routing with Constraint Logic Programming and Delay. 9th I.C. on Industrial Applications of AI. Gordon and Breach. (1996) 217-231.

Using a Genetic Algorithm to Select Parameters for a Neural Network That Predicts Aflatoxin Contamination in Peanuts

C. E. Henderson[a], W. D. Potter[a], R. W. McClendon[b], G. Hoogenboom[b]
Artificial Intelligence Center[a], GSRC 111
Bio & Ag Engineering Department[b]
University of Georgia, Athens, GA 30602
USA
dpotter@ai.uga.edu, rwmc@bae.uga.edu, gerrit@tangine.bae.griffin.peachnet.edu

Abstract. Aflatoxin contamination in crops of peanuts is a problem of significant health and financial importance, so it would be useful to develop techniques to predict the levels prior to harvest. Backpropagation neural networks have been used in the past to model problems of this type, however development of networks poses the complex problem of setting values for architectural features and backpropagation parameters. Genetic algorithms have been used in prior efforts to locate parameters for backpropagation neural networks. This paper describes the development of a genetic algorithm/backpropagation neural network hybrid (GA/BPN) in which a genetic algorithm is used to find architectures and backpropagation parameter values simultaneously for a backpropagation neural network that predicts aflatoxin contamination levels in peanuts based on environmental data.

INTRODUCTION

Aflatoxin contamination in peanuts is a well-known agricultural problem of health concern (U.S. Food and Drug Administration Center for Food Safety and Applied Nutrition, 1992). This concern produces financial pressures for growers, resulting in efforts to develop methods to predict contamination levels expected for crops. A relationship between environmental factors and aflatoxin levels has been identified (Thai et al., 1990). In particular, factors such as soil temperature and drought are believed to have a strong effect on aflatoxin levels in peanuts (Cole et al., 1985). Therefore there is interest in developing a tool that predicts aflatoxin levels based on environmental data. Such a tool could be used to guide crop management decisions and to select a harvest date that minimizes contamination levels.

Artificial neural networks (ANNs) provide a methodology for constructing predictive tools and have been used in prior efforts for the aflatoxin problem. However there are difficulties inherent to the process of constructing neural network models. To make accurate predictions, a neural network requires constructing an appropriate architecture and setting values for the connection weights. The relationship between architecture and network performance is poorly understood, and searching for the best one is a heuristic task. This search is often conducted by hand in a trial and error fashion and can be tedious and error prone. The task of searching for connection weights, known as learning or network training, is also challenging. The backpropagation algorithm is an effective technique for network training, but the algorithm itself requires setting parameters for momentum and learning rate. So backpropagation creates a new problem while solving another. A technique for algorithmically locating and evaluating neural network parameters would therefore be useful. Such a search will be termed an automated method here.

Genetic algorithms provide a way to search large, poorly understood spaces, and have been used successfully to set neural network parameters. By linking the global search capabilities of the genetic algorithm with the modeling power of artificial neural networks, a highly effective predictive tool for the aflatoxin problem can be constructed.

This paper describes the design and implementation of a genetic algorithm/backpropagation hybrid (GA/BPN) that is used to produce neural networks for the aflatoxin problem. The goal of this work is to devise a system that uses a genetic algorithm to find parameters for a backpropagation neural network that models the aflatoxin contamination problem. GA/BPN results will be compared to those of Henderson (1997) to determine the effect of adding the genetic enhancements to common backpropagation.

BACKGROUND

Many efforts, beginning in the late 1980's, have employed genetic algorithms to evolve parameters for neural networks. Representation is an important issue for such a system and may be generally grouped into direct and indirect encodings. Miller et al. (1989) evolved internode connectivity for a feedforward neural network with a fixed number of layers and hidden nodes. Connectivity was represented using a binary string in which each bit corresponded to a pair of nodes in the network, and all possible pairs were represented. Miller et al. (1989) used a specialized crossover operator that ensured that bits were copied in blocks, each corresponding to all of the outgoing links of a single node. It was believed that such groups of links were functional units and that preserving them would aid convergence. The mutation operator was ordinary, flipping bits according to a probability that was set to a low value. The selection operator used a common fitness proportionate scheme. The fitness function decoded a chromosome to an architecture, trained the network using backpropagation for a preset number of epochs, then returned the sum of the squares of the errors on the training data set for the last epoch. The system was tested for the XOR problem, a real valued four-quadrant problem, and a pattern copier that contained fewer hidden nodes than inputs. The approach was effective, but the three problems were each quite simple and did not provide a full test of the computational power of the method. (Mitchell, 1996)

Marti (1992) used a direct encoding in a novel way, including two genetic systems. The first system, called the inner one, was used to generate neural network connectivity based on a representation, while another, outer, system evolved the representation itself.

Indirect encodings were developed to overcome some of the shortcomings of direct ones. Because directly encoded chromosomes map directly to a phenotype, they grow in size at a rate equal to that of the phenotype. This may make direct encodings computationally impractical for large problems. Indirect encodings, on the other hand, include small chromosomes that decode to larger networks via sets of generative rules. Kitano (1990) provided an elegant example of an indirect representation known as a grammatical encoding. By this scheme, chromosomes encoded a set of grammar expansion rules. Such chromosomes were decoded by repeatedly applying the rules until a network was produced. Fitness was deduced by decoding and training via backpropagation, similar to Miller et al. (1989), described above. It was found that this encoding was effective for a number of low complexity problems (Mitchell, 1996). Examinations of strengths and weaknesses of a number of encoding schemes may be found in Roberts and Turega (1995) and Balakrishnan and Honavar (1995).

A number of efforts have evolved values for backpropagation parameters for neural networks. Harp et al. (1989) used a genetic algorithm to set the topology and learning rate for a backpropagation neural network. An innovation of the approach was the inclusion of a mechanism by which the learning rate varied over time and among groups of nodes. The design was found to be effective on several simple problems, including XOR. The primary criterion by which networks were evaluated was speed of convergence to an acceptable solution.

Schaffer et al. (1990) also used a genetic algorithm to set the topology and learning parameter for a backpropagation neural network. The search included the momentum parameter and the range of initial weights. Networks were trained for a 4 bit coding problem for a maximum of 200 training epochs. It was found that for the problem, low learning rates (0.25 and 0.5) and low initial weights ranges (-0.125 to 0.125 and 0.25 to 0.25) were most effective.

There have been several efforts that used neural networks to model the aflatoxin prediction problem that was described in the Intoduction. The focus of this paper is on setting backpropagation and network architecture parameters, so this aspect of prior efforts will be examined most closely hereafter. Parmar et al. (1997) trained three layer feedforward neural networks using backpropagation for the aflatoxin problem. Data was collected and allocated to a training and test set, and results were compiled using a range of values for the number of nodes in the hidden layer. Pinto (1996) trained networks for the same problem, varying learning rate and momentum in addition to the number of hidden nodes. Pinto (1996) used the peanut growth data compiled by Parmar et al. (1997), but created a validation data set in addition to training and test sets. He also created two network models, one (model A) for all of the data that was used by Parmar et al. (1997) and one (model B) that used only data from undamaged peanuts. Results were generated for both models across training, test, and validation data sets. Henderson (1997) used a common backpropagation implementation to reproduce the work of Pinto (1996). Both studies used identical data sets. The only difference in the work was that Pinto (1996) used a commercial product to train networks, while Henderson (1997) used common backpropagation. It was concluded in Henderson (1997) that the common algorithm produced results that appeared to be generally representative of the performance of backpropagation when parameters are set by hand and would therefore be useful for a performance comparison study.

MATERIALS AND METHODS

A network design that has been used for aflatoxin prediction is the three layer, feedforward neural network that uses backpropagation as the learning algorithm, and such a network was chosen for this experiment. Hereafter "network" or "neural network" will refer to a network of this type.

Many times, the development of a model is started by gathering sets of patterns that are generally representative of the problem that is to be modeled. This data is allocated to a training set, test set, and validation set. The training set can be used by backpropagation as a guide for making adjustments to the weights of the network. It is useful to employ the test set to decide when to stop training the network. The validation set serves to evaluate the performance of the network on new data after training is complete. The described approach to data preparation was taken for this experiment.

The process of developing a neural network model for a real world problem can be a time consuming, complex task. Many parameters must be set, and the effect that chosen values have on network performance is often poorly understood. The number of nodes in the hidden layer is one example of a problematic parameter. The problem of setting the values for the weights is handled using the backpropagation algorithm. However using backpropagation introduces more parameters, including learning rate and momentum. It is known that learning rate and momentum can affect the quality of the final network, but there is no clear way to determine what values will prove to be best. These difficulties often necessitate a try it and see approach to neural network development, in which the investigator manually sets values for these parameters, observes how effective they prove to be, adjusts according to some heuristic, and tries again. This technique can be ineffective, because only a few parameter sets may be examined. Also, the investigator will rarely spend much time trying values that appear unpromising at first, even though these values could prove to be optimal ones. For these reasons, it is desirable to employ an automated search technique that avoids local minima while performing a global search of the parameter space. The genetic algorithm is appropriate for this problem and its use is described herein. Other effort to set backpropagation parameters with genetic algorithms may be found in Harp et al. (1989), Schaffer et al. (1990), and Belew et al. (1990).

Genetic algorithms are search methods based on the mechanics of natural selection and natural genetics (Goldberg, 1989). They are effective global search techniques that avoid many of the shortcomings of more traditional searches such as hill climbing. In particular, hill climbing is highly susceptible to getting stuck in local minima. A genetic search explores many points in the search space simultaneously, thus blanketing regions and providing a mechanism for avoiding local minima. Genetic algorithms do not rely on local information to guide the search, but require only a function that can assign a rough measure of merit to a candidate solution. Genetic algorithms have proven effective for locating high quality solutions among large, poorly understood search spaces. These strengths match well with the difficulties of configuring neural networks, so the genetic algorithm was selected to locate network parameter values for this experiment.

There are at least three major elements of a traditional genetic algorithm: Representation, operators, and the objective function. The representation provides a specification for storing the information that is to be evolved toward a solution. The genetic algorithm maintains a pool of potential solutions, which are built according to the choice of representation. Operators provide the machinery by which pools of chromosomes are manipulated. Common operators for a genetic algorithm include a selection scheme, a crossover operator, and a mutation operator. The selection scheme is a method for choosing chromosomes from a pool to be used as parents for a new pool. Crossover is a mechanism for combining parent chromosomes to produce offspring. Mutation modifies chromosomes at random in order to produce a continual source of genetic diversity.

The objective function is of key importance to a genetic algorithm implementation, because it is the sole source of information about the value of candidate solutions. Given a chromosome, the objective function returns a number indicating merit, and this measure of merit is used by the selection scheme to determine which solutions survive and which ones perish.

Data Collection and Preparation

The following data collection descriptions are taken from Pinto (1996). The data that was used for this experiment was obtained from the United States Department of Agriculture Agricultural Research Service (USDA ARS) National Peanut Research Laboratory (NPRL) at Dawson, Georgia. Measurements were taken from florunner peanuts that were grown in environmentally controlled stands. Following harvest, all peanuts were analyzed for aflatoxin contamination. Aflatoxin levels for entire stands were determined using the weighted average of the grade values.

Data sets were available for the years 1985 through 1995. Each observation consisted of the aflatoxin and environmental values for a specific plot and season. Environmental values included length of drought stress period (days), mean soil temperature (degrees Celsius), crop age (days), and accumulated heat units (degrees Celsius days). Drought stress was the number of consecutive days of drought conditions, while mean soil temperature was the mean temperature of the soil during this period. Crop age was the number of days from planting to harvesting. Accumulated heat units (AHU) was the accumulation of heat above 25 degrees Celsius during the drought period. This value was calculated by the following equation:

$$\text{AHU} = (\text{mean soil temperature} - 25) * \text{length of drought stress period}$$

This calculation was taken from Parmar et al. (1997) and was also used by Pinto (1996). These four environmental factors (drought duration, mean soil temperature, crop age, and accumulated heat units) are used as inputs for the neural networks developed in this project.

Because it was observed that the inclusion of damaged peanuts in the data introduced a great deal of noise, Pinto (1996) developed two neural network models. The first (model A) included data for both damaged and undamaged peanuts, while the second (model B) included only measurements for undamaged peanuts. The available data was used to produce two pools, one for model A and one for model B. For each of these pools, the data was sectioned into training, test, and validation sets. So that meaningful comparisons could be made between results, the current project used the same data sets as Henderson (1997), that were taken from Pinto (1996).

The purpose of this project was to compare the effectiveness of a genetic search for network parameters to traditional backpropagation. It is therefore necessary to have comparison results from models of the aflatoxin problem that did not include a genetic search. Henderson (1997) developed a common backpropagation neural network models using the same data as the current project. Therefore, this data will be used as a standard of comparison to evaluate the effectiveness of the GA/BPN approach.

Networks were evaluated by comparing predictions against target values for the patterns in the data sets and were developed separately for model A and model B data. r^2 values, the square root of mean squared error (RMSE), and the mean absolute error (MAE) were used as metrics. These values were calculated for the two models for each of the data sets.

MODEL DEVELOPMENT

As described previously, two aflatoxin models were developed, dubbed A and B. A three layer, feedforward network has been used for many similar problems and was chosen as the modeling tool for the aflatoxin problem. Backpropagation was chosen to be the learning algorithm.

There are a number of parameters that must be assigned values in order to develop a backpropagation neural network model of high quality. Such parameters include number of layers, connectedness of layers, number of hidden nodes, learning rate, and momentum. There is no clear-cut procedure for choosing values for these parameters that will produce the best network model, hence neural network development is often more art than science. Some efforts to develop automated techniques to set backpropagation parameters are Harp et al. (1989), Schaffer et al. (1990), and Belew et al. (1990). The goal of this project was to employ the search power of the genetic algorithm to choose settings, however problem complexity precludes evolving values for all of these parameters. The number of hidden nodes, learning rate, and momentum were chosen to be evolved, because it was observed in Pinto (1996) and in Henderson (1997) that these three parameters have a strong impact on the quality of the backpropagation networks that are produced for the aflatoxin problem. An architecture including three fully connected layers was effective in Pinto (1996) and in Henderson (1997), so a similar one was used here.

To employ the genetic algorithm to search for neural network parameters, there are a number of design issues that must be addressed. These issues are related to the representation, selection scheme, crossover scheme, mutation operator and the objective function. Choice of representation dictates many design decisions and should be examined first. The items to be represented are learning rate, momentum, and number of hidden nodes, which may be easily represented as real numbers. The network parameters to be evolved were therefore represented as strings of real values.

A chromosome consists of three real numbers constrained to lie between 0.0 and 1.0. Decoding proceeds as follows: The first and second numbers are used to produce the learning rate and the momentum rate, while the third number is scaled by a preset maximum allowable number of hidden nodes, converted to an integer, and used to set the number of hidden layer nodes. To reduce computational demands, an upper limit of 21 was placed on the number of hidden nodes for the networks examined in this experiment. The most effective networks in Pinto (1996) and Henderson (1997) contained no more than 14 hidden nodes, so the limit of 21 is expected to be sufficiently high.

A crossover scheme was selected based on empirical results. Several tests were conducted using each of uniform, two-point, and one-point crossover. Single point crossover consistently produced better convergence and better final solutions. So single point crossover was chosen.

Because mutation plays an unusually important role for this implementation, great care was taken in its development. Consider a hypothetical, highly fit chromosome that has just been produced by crossover. Because fitness is high, presumably the real values in the string are close to optimal ones. This may not always be the case, but it is the assumption that this will often be the case that underpins the power of the genetic algorithm. A traditional mutator might traverse the bits in the chromosome, flipping an occasional bit based on a biased coin toss. However, with real values this is not desirable. Flipping a bit in a floating point number will frequently result in new values that are much greater or smaller than the original. Because the real numbers are assumed to already be close to optimal values, large jumps are undesirable. For this reason, a customized mutator was implemented for use with the real valued strings. The mutator picks a new value based on a distribution around the value being mutated. In effect, it adds or subtracts a small number from the allele. Hence, the mutator should provide significant new genetic diversity, but will not destroy valuable alleles by changing them drastically. Because the implementation relied heavily on the mutator to produce diversity, a mutation rate was chosen that was larger than usual, 0.08.

For a given genotype, the fitness of its phenotype is not unique for this problem. This is because the objective function employs backpropagation, which relies on an initial random seeding of the weight space to provide a search starting point. Hence, a genotype that can lead to an excellent network may receive a poor fitness rating due to unlucky initial weight assignments. The initial weights may be such that backpropagation gets stuck in a local minimum. To avoid this problem it is essential to evaluate promising genotypes multiple times. Luckily, the genetic algorithm provides just such a mechanism, selection. Many times selection is merely a way for a genetic algorithm to choose the most promising genotypes to use as parent chromosomes, but in this case there is an added bonus. If selection has no bias against selecting identical chromosomes from a pool, then the genotypes will get evaluated more than once. After a few generations, there will be many copies of the same fit chromosome in a population. Typically, this points to premature convergence and is to be avoided. However, recall that an unusual mutator and a high mutation rate were employed in this case. So even with a homogenous population, there is still strong pressure toward new diversity. For this reason, selection is of unusual importance to this implementation, as it provides a mechanism by which potentially strong genotypes may be evaluated more than once. Standard roulette wheel selection provides all of the features that are desirable and was chosen.

The objective function provides all of the evolutionary pressure for a genetic search, so its design is of key importance. The problem under investigation was to develop a genetic algorithm that would evolve parameters that when used with backpropagation would lead to highly effective networks. So the measure of fitness for a chromosome was chosen to be a function of its effectiveness with backpropagation.

There are some marked differences between training a neural network using simple backpropagation and training one via the GA/BPN. An important consideration regards deciding how to evaluate and compare results. A common method for training a network via backpropagation proceeds as follows. The data is broken into training, test, and validation data sets, and backpropagation modifies weights using error calculated over the training set. Occasionally, backpropagation checks its performance against the test set. When performance on the test set stops improving, the algorithm stops training and returns as a solution the one network that performed best on the test data set. Various measurements of accuracy are then calculated for the network for each of the data sets. Hence, locating the solution is straightforward. The solution network is the one that performed best on the test set and it is evaluated in terms of its performance on the validation set.

Note, however, that the genetic algorithm and simple backpropagation network produce solutions in different ways. Backpropagation incrementally improves the network on the training set, using the most recent weight assignments as a starting point for each training event. It is expected that the network will show concomitant improvement on the test set as small training set improvements are made. Hence, backpropagation moves the network along a performance continuum, exploiting local information all the way, until a minimum is encountered. The genetic algorithm, however, exploits no local information, relying entirely upon one number (training set mean absolute error) to provide training pressure. The result is that the genetic algorithm may not proceed smoothly along a continuum toward a solution as backpropagation does. Instead, it will produce pools of networks that are increasingly good fits for the training data, but it is difficult to determine which of these networks provides the best overall solution. Therefore a solution selection scheme was necessary to keep the best networks as they were encountered. Because any given network in a pool had a chance of being the best ever, it was necessary to consider each chromosome of every generation. For this reason, the solution selection mechanism was included as an addition to the objective function, which accesses all chromosomes.

The selection scheme was designed on the assumption that the network that performs best overall on the training and test data sets would provide the best predictions for new patterns. However, this requires a procedure for determining what best means. Clearly, it is desirable to have a network that has a low total for the two error values. However, it is undesirable to keep a network that has an extremely low value for one set, but not the other. Such a net constitutes an overfit for the one data set, and may not be robust. For this reason, the second criterion of step 2 was included, namely that the errors must be close in value. Step 2 requires a hard coded tolerance value that specifies how close the two error values must be to each other. This tolerance was determined based on the difference observed in high quality networks from previous studies and was set to an absolute value of 15.0 units. The genetic search was set to terminate after a predetermined number of generations, and the stored best network was returned as the solution. The networks found for each of the two models (both A and B) were then evaluated in terms of mean absolute error, root mean squared error, and r^2 value for each of the training, test, and validation data sets. These values were compiled as final results for the experiment.

RESULTS

Networks were trained for model A and model B, and performance measures were compiled across the data sets. These results were developed using a genetic algorithm/backpropagation neural network combination, and therefore will be denoted as the GA/BPN results. Note that the GA/BPN system produced one best network for each model. GA/BPN results are compared to those of Henderson (1997). Henderson (1997) values are called Simple BPN and supply a baseline performance measure for simple backpropagation for the aflatoxin problem.

GA/BPN produced networks that outperformed those of Simple BPN for both model A and model B. Hence, the GA/BPN approach is probably the more effective of the two techniques for the aflatoxin problem. Because aflatoxin prediction is similar to many other predictive problems, it is likely that the superiority of the GA/BPN will apply to other problems as well.

It was mentioned previously that network performance on the validation data sets was not high. GA/BPN made accurate predictions for all of the validation data except for a few marked exceptions. Both model A and model B displayed poor performance on the validation data sets relative to training and test sets. For model A, training and test set r^2 values of 0.85 and 0.89 were attained, but the value for the validation set was only 0.15. Validation results for model B are also low, although not to the same extent as model A. For model B, r^2 values of 0.77 and 0.70 were produced for training and test data, but a value of only 0.51 was attained for validation data.

A possible explanation for the low validation set performance is that the data sets are noisy. Scatter plots of predicted versus target values for validation data for the GA/BPN approach are enlightening, showing good fits for all but a few patterns. Predictions for these few patterns, however, are often wildly inaccurate. This is a strong indicator that these errors are the result of noisy data. Because there are a small number of validation data points, these few highly inaccurate predictions have an strong detrimental effect on the performance metrics for the set. Future researchers that make use of this data might consider preprocessing it in some way that will reduce noise. Of course, an even better solution would be to gather a larger number of patterns, so that the noise may be statistically absorbed.

Comparison of validation data set performance between model A and model B supports the assertion of Pinto (1996) that damaged peanuts tend to produce noisy data. Recall that model A used of all of the available aflatoxin data, while model B only used data from undamaged peanuts. Training and test set performances for model A and model B are similar (model A r^2 values of 0.85 and 0.89 compared to model B values of 0.77 and 0.70). However, model B shows a large validation data set performance increase (model A r^2 of 0.15 compared to model B value of 0.51). This suggests that there was indeed less noise in the model B data.

Possibly the greatest value of the GA/BPN approach lies in the ease with which it may be used to find good networks. The only parameters that need to be set are crossover rate, mutation rate, and pool size. This may not sound better than normal backpropagation, which also requires three parameter values (learning rate, momentum, and number of hidden nodes). However, the genetic search was fairly insensitive to these settings, finding good networks for every set of parameter values that was tried. Simple BPN, on the other hand, was highly sensitive to parameter settings and would find only extremely poor networks for the majority of values.

GA/BPN proved to be more effective than Simple BPN for training networks to predict aflatoxin contamination levels in peanuts. It is expected that this superiority will apply to other problems as well. GA/BPN was also much easier to use and required little human interaction. This ease of use should pay off for developers by saving both time and effort. Also, more faith may be placed in the value of the parameters found by GA/BPN than in those located via traditional methods, because GA/BPN has been shown to produce the higher performance networks.

REFERENCES

Balakrishnan, K., and V. Honavar 1995. Properties of genetic representations of neural architectures. Proceedings World Congress on Neural Networks, 807-813.

Belew, R. K., J. McInerney, and N. N. Schraudolph. 1990. Evolving networks: Using genetic algorithms with connectionist learning. CSE technical report CS90-174. La Jolla, CA: University of California at San Diego.

Cole, R. J., T. H. Sanders, R. A. Hill, and P. D. Blankenship. 1985. Mean geocarposphere temperatures that induce preharvest aflatoxin contamination of peanuts under drought stress. Mycopathologia. 91(1): 41-46.

Goldberg, D. E. 1989. Genetic algorithms in search, optimization and machine learning. Reading, MA: Addison Wesley.

Harp, S. A., T. Samad, and A. Guhu. 1989. Towards the genetic synthesis of neural networks. Proceedings of the Third Int. Conference on Genetic Algorithms. 360-369.

Henderson, C. E. 1997. Using genetic algorithms to evolve neural networks. Master's Thesis. Artificial Intelligence Center. The University of Georgia.

Kitano, H. 1990. Designing neural networks using genetic algorithms with graph generation system. Complex Systems 4. 461-476.

Marti, L. 1992. Genetically generated neural networks II: Searching for an optimal representation. Proceedings 1992 IEEE Int. Conf. on Neural Networks. 221-226.

Miller, G. F., P. M. Todd, and S. U. Hegde. 1989. Designing neural networks using genetic algorithms. Proceedings Third Int. Conf. on Genetic Algorithms. 379-384.

Mitchell, M. 1996. An introduction to genetic algorithms. Cambridge, MA: The MIT Press.

Parmar, R. S., R. W. McClendon, G. Hoogenboom, P. D. Blankenship, R. J. Cole, and J. W. Dorner. 1997. Estimation of aflatoxin contamination in preharvest peanuts using neural networks. Transactions of the ASAE. 40(3): 809-813.

Pinto, C. S. 1996. Prediction of aflatoxin contamination in peanuts using artificial neural networks. Master's Thesis. CS Department. The University of Georgia.

Roberts, S. G., and M. Turega. 1995. Evolving neural network structures: An evaluation of encoding techniques. Artificial neural nets and genetic algorithms: Proceedings of the International Conference in Ales, France. 96-99.

Schaffer, D., R. A. Caruana, and L. J. Eshelman. 1990. Using genetic search to exploit the emergent behavior of neural networks. Emergent computation. 244-248.

Thai, C. N., P. D. Blankenship, R. J. Cole, T. H. Sanders, and J. W. Dorner. 1990. Relationship between aflatoxin production and soil temperature for peanuts under drought stress. Transactions of the ASAE. 33(1): 324-329.

U. S. Food and Drug Administration Center for Food Safety and Applied Nutrition. 1992. Foodborne Pathogenic Microorganisms and Natural Toxins. Washington, D. C.: U. S. Food and drug Administration.

Interacting with Articulated Figures within the PROVIS Project

Hervé Luga †, Olivier Balet †/‡,Yves Duthen †, René Caubet †
† I.R.I.T., Paul Sabatier University, 118, Route de Narbonne, 31062
Toulouse Cedex, France
e-mail: { luga I balet }@irit.fr
‡ CISI, 13, Rue Villet, Z.I. du Palays, 31029 Toulouse, France
e-mail: Olivier.Balet@cisi.cnes.fr

Abstract: The main goal of our work is to allow designers to replace physical mock-ups by virtual ones in order to test the integration and space requirements of satellite components. To this end, PROVIS combines a "desktop VR" interface with an interactive 3D display. This paper describes how PROVIS allows users to interact with flexible or articulated objects, for testing maintenance procedures and component accessibility, thanks to an original use of genetic algorithms.

1. Introduction

Nowadays, physical mock-ups are used in a wide range of activities (aerospace, automotive manufacturing or architecture). They still remain the main solution for testing integration, space requirements and accessibility of their equipment. Either built of paper, cardboard and glue or, in a more "modern" way, of wood and new materials, they still constitute tedious, time consuming, costly solutions. Moreover, physical mock-ups, once realised, are neither flexible nor easy to manipulate. That is why CNES, the French space agency, CISI and IRIT jointly launched the PROVIS research project in 1995. The main aim of this project was to find solutions which would allow satellite designers to create, manipulate and study their models by using digital mock-ups called prototypes.

To increase the efficiency of this experiment and offer as user-friendly an interface as possible, we had to use leading-edge technological concepts, particularly a "semi-immersive virtual reality" type interface associated with high-level commands. Actually, we mainly wanted to save users the trouble to study a new system for prototyping tasks.

This paper presents an original and innovative way to interact with articulated objects such as wires, flexible pipes and human bodies. Genetic algorithms[1] were used to define the behaviours of articulated objects thus replacing known methods[2] such as inverse kinematics. Lastly, the use of the same algorithms to study component accessibility and prototype dismantling by planning trajectories will be discussed.

2. Presentation of PROVIS

PROVIS was designed to allow non-expert users to build virtual prototypes which can be used to test the integration and space requirements of equipment. It allows to define and operate kinematics between the mobile parts of a system while exploring and expanding the on-line documentation. In addition to an intuitive use which derives from strong interactivity, the main value of this package is that it allows a qualitative assessment of the prototype components (integration, space requirements, weights, electric consumption, etc.). The 3D graphical representation is actually the visible part of a composite component featuring: physical attributes (weight, etc.), functional attributes (object belonging to a subsystem, etc.), documentary attributes (URL links, text and audio comments, etc.). The graphical envelope is an access metaphor to other data which are accessed by physically designating a particular component. Furthermore, several designers located on remote sites can work with PROVIS on the same shared prototype and confront their conceptions.

3. Interacting with articulated objects for prototyping and maintenance

PROVIS allows users to manipulate 3D objects in a natural way. Although, more complex behaviours had to be introduced to manipulate complex objects such as wires or flexible pipes. In our system, these objects are modelled as articulated chains defined by tubes, joints and constraints (flexibility, etc.). Unlike the motion of passive systems like falling objects or elementary components, the motion of articulated objects, human or robot arm for the purpose of manipulation are motions with intentions[4]. Designers had to be offered two kinds of interaction concepts :

Direct manipulation: The user tracks a 3D pointer to interact with an articulated object, grasping either the end or any other part of the object.

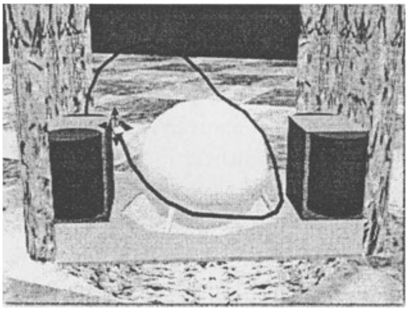


Fig. 1. Direct manipulation of an articulated object

Task-level manipulation: The user chooses a wire, defines a source and a target, the favourite zones, the zones not allowed and launches the task resolution. Then, the wire plans its collision-free path which respects the specified constrains. Figure 2 shows the various parameters which have been interactively defined by the user. The red sphere shows a zone which must be avoided while planning the wire path. The clip is a zone which the wire has to pass through. Note that the clip used is just a position constraint and that we also provide users with another kind of clip which represents both position and rotation constraints. Whereas the solution presented involved implied the clip rotation, it respects the position constraint.

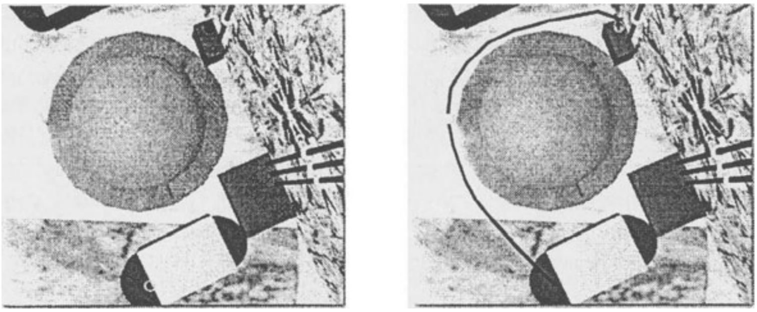


Fig. 2. (a)Defining the task parameters, (b)Solution computed

Once the prototype realised, designers wanted to test the maintenance procedures to check whether or not the components could be easily removed from the prototype. This verification stage is one of the most important phases of every prototyping project. It is one of the main improvement on classical maintenance tests with physical mock-ups. At this stage, any error would imply a cost rise for the satellite production. That is why we allow the user to check any virtual component accessibility introducing arm and tools models (figure 3) in the virtual environment.

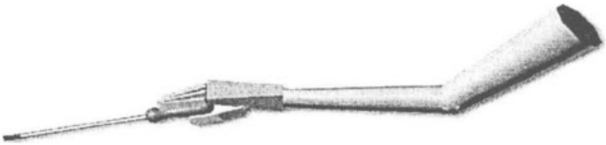


Fig. 3. Arm and tool 3D models

Then, the operator selects a component and verifies whether the tool he uses can reach and dismantle it from the prototype structure. Afterwards, he checks whether this component can be extracted from the prototype, i.e. there is a valid path for the component to reach out of the prototype. The operator's arm and the tool he holds are modelled like any other articulated object. That is the reason why the accessibility tests can be seen as the previously described task-level manipulation.

Our solution, based on genetic algorithms, is for the time being one of the most useful ones to seriously study prototype dismantling. Several other experiments dealt with this kind of problems by integrating the user in the simulation: the user wears 3D sensors and datagloves to test by himself whether he can reach and remove the components. While portable force feedback devices cannot manage the whole operator's body, none of these solutions can take into account the real muscular strain and those body parts can lean on components during the operation.

Several methods have already been proposed to manage articulated figures[2]. Based on inverse kinematics[5] or inverse dynamics models, for example, these methods never offer an intuitive way to define internal and global constraints. Moreover, people have to use them in association with a motion planner in order to achieve task assessment. In conclusion, there is no solution to manage homogeneously inverse kinematics, local constraints (on each Degree Of Freedom), global constraints (for instance, the fragility of a joint can prevent the wire from hanging out of a satellite structure), external constraints (collision detection, favourite zones, etc.) and motion planning. Therefore, we propose an original use of genetic algorithms in order to define behaviours of articulated objects so that could be managed in a very homogeneous way, interactions with fully constrained articulated objects.

3.1 Genetic algorithms

These search and optimisation algorithms were introduced by Holland[6]. They use as a model the natural evolution process to provide a robust search technique in complex environments[1]. This method mimics the environment pressure on a population of data structures which represent the genetic codes of the solutions to the problem. Each individual of the population, called chromosome, is evaluated against the environment and receives a payoff. This measure, provided by a fitness function, has to estimate the ability of an individual to solve the problem.

3.2 Adaptation to our problems

Our aim is to generate linked objects which can automatically adapt themselves to their environment which consists of a set of constraints. This set includes the distance between a target and the articulated object end, as well as inclusion and exclusion zones (i.e. favourite zones). For instance, exclusion zones can be hazard zones which must be avoided in order to provide a reliable construction. Moreover, inclusion zones can represent forced passages inside the prototype. Please note that we will limit our examples to 2 dimensional problems. Extending the problem to 3D can be very easily achieved with our method.

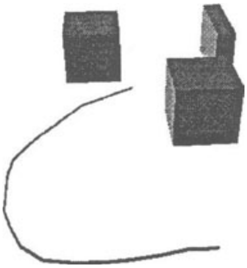
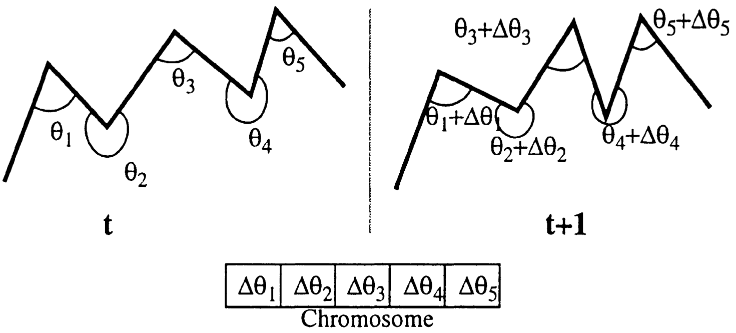


Fig. 4. A simple 2D articulated object

At each time step, we must compute each joint position. These positions are described by the relative angles $[\theta_1^t, \theta_2^t, \ldots, \theta_N^t]$ between each component of the articulated object. Therefore, our algorithm could be considered as a function $F: [\theta_1^t, \theta_2^t, \ldots, \theta_N^t, E^t] \rightarrow [\theta_1^{t+1}, \theta_2^{t+1}, \ldots, \theta_N^{t+1}]$ which computes the new positions $[\theta_1^{t+1}, \theta_2^{t+1}, \ldots, \theta_N^{t+1}]$ of the joints according to both the previous ones and the environment state E^t. Therefore, it has to compute the differences $\Delta\theta_i$ between the current position θ_i^t and the new position θ_i^{t+1}. So, our algorithm is a function $F: [\theta_1^t, \theta_2^t, \ldots, \theta_N^t, E^t] \rightarrow [\Delta\theta_1, \Delta\theta_2, \ldots, \Delta\theta_N]$ and the new articulated object position could be computed using $\theta_i^{t+1} = \Delta\theta_i + \theta_i^t$.


θ_1
θ_2
θ_3
θ_4
θ_5
t
$\theta_3+\Delta\theta_3$
$\theta_5+\Delta\theta_5$
$\theta_1+\Delta\theta_1$
$\theta_2+\Delta\theta_2$
$\theta_4+\Delta\theta_4$
t+1
Chromosome

$\Delta\theta_1$	$\Delta\theta_2$	$\Delta\theta_3$	$\Delta\theta_4$	$\Delta\theta_5$

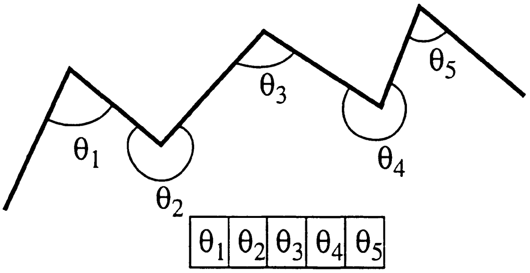
Fig. 5. Definition of a chromosome using position differences

If we consider that each $\Delta\theta_i$ could take V distinct values then, card(V)=V^n . For example, if each could take values into [-0.1..0.1] with a 10^{-3} precision and if our object is composed of 20 joints then, the state space cardinal is 200^{20}= 1.04857x10^{46}.

As stated before, we now have to define a chromosomal representation which represents the different configurations of our articulated object and a fitness function able to evaluate the pertinence of the corresponding solution trial.

3.3 The chromosome representation

Each chromosome represents a configuration of an articulated object. It is composed of the list of all relative angles which represent the degrees of freedom of the object. In opposition to the original genetic algorithm definition which uses a bitstring coding for chromosomes, we chose to use a real-number chromosomal representation. In fact, this representation is more efficient to solve our problem because the chromosomes do not need to be decoded before computing their fitness. Consequently, we have to define new genetic operators on real number lists. An example of such a redefinition is the crossover operator. When two chromosomes are mixed according to genetic algorithm definition, the resulting individuals are composed of a part of each parent and of an algebric average of some randomly chosen values in both parents.



θ_1	θ_2	θ_3	θ_4	θ_5

Fig. 6. A 2D chromosome

3.4 The fitness function

We now have to define the fitness function. This function computes the adaptation of a chromosome to its environment.

$$fit(Chromosome) = fit(\theta_1^t, \theta_2^t, \ldots, \theta_N^t, \Delta\theta_1, \Delta\theta_2, \ldots, \Delta\theta_N, E^t) \rightarrow \Re$$

It must take into account, at a given time t, the current object configuration, the relative angles proposed by the chromosome and the environment state which includes the constraints imposed to the object. This function represents the environment pressure on the population and does not have to be too selective to avoid prime convergence to a sub-optimal solution. Moreover, it must describe the real differences between two chromosomes to provide a convenient measure of the adaptation.The main advantage of our solution is to offer a very easy way to constraint articulated object movements.
Examples of fitness functions:
Object tracking: "The end of an articulated chain has to track a pointer à la inverse kinematics."

$$fit(Chromosome) = -\left\|\overrightarrow{end - pointer}\right\|$$

Object tracking + Zones avoidance: "The end of an articulated chain has to track a pointer while avoiding collisions between the chain and the Zi zones."

$$\begin{aligned} fit(Chromosome) = \quad & -\left\|\overrightarrow{end - pointer}\right\| \\ & -\eta_{collisions} \cdot \sum_{Z_i} \text{nbCollisions}(object, Z_i) \end{aligned}$$

Object tracking + Zones avoidance + Zones favouritism: "The end of an articulated chain has to track a pointer while favouring the passage in the Fi zones and avoiding collisions between the chain and the Zi zones."

$$\begin{aligned} fit(Chromosome) = \quad & -\left\|\overrightarrow{end - pointer}\right\| + \\ & \lambda_{collisions} \cdot \sum_{F_i} \text{nbCollisions}(objet, F_i) \\ & -\eta_{collisions} \cdot \sum_{Z_i} \text{nbCollisions}(object, Z_i) \end{aligned}$$

As shown in the previous examples, it is very easy to add or remove constraints to our system. We could have constrained in the same way the different degrees of freedom of the articulated object by limiting each angle variation interval.

3.5 Results

We have made several benchmarks in order to verify whether or not our solution was suitable for real time interaction. Two kinds of tests were carried out with the same wire made of 20 joints (3 DOFs per joint).

1. The end of the wire had to track the user's 3D pointer. Each joint had a flexibility constraint in order to respect its physical model: The tracking was achieved in real time (less that 10 msec latency) for "normal" movements.

2. 100 target positions the wire had to reach autonomously were choosen randomly. Then, we measured the average time the wire took to plan his trajectory and reach the

target (the velocity of the wire movement was constant during the whole tests). This average time was processed for the same target positions and for different environments.

Direct manipulation: Each test shows the ability of our algorithm to provide with a realtime system. The computation of each move is achieved in less than 0.1 second on a R4000 SGI workstation.

Task level manipulation: Our algorithm generally compute solutions in ess than 10 seconds in all of our experiments and therefore could be integrated as an interface module which computes routing in backgroud while the user could achieve some other tasks.

3.6 Analysis of a run

We can take an insight into the algorithm by looking at the maximum (figure 7) and mean fitness during a run. Presented results are extracted from a realtime 2D simulation where the articulated object has 20 degrees of liberty. The used fitness function permits to follow the user clone avoiding collisions with objects. We choose a strong contraint on the collision avoidance by setting $\eta_{\text{collisions}} \gg \text{Maximum Distance(pointer,end)}$. A fitness lower than -1000 will correspond to a collision. These simulations shows 270 moves with 6 genetic generations between each move. The population size is 60 chromosomes.

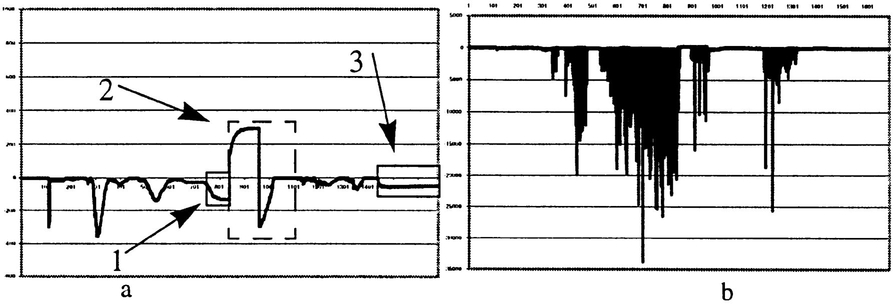


Fig. 7. Maximum (a) and mean (b) fitness on a real experiment

First, we can verify that the whole maximum fitness curve is above -1000. This shows that none of the articulated object moves generates a collision. Moreover, the curves stays mainly near 0 which represents the minimal distance between the pointer and the end of the articulated object. We can focus on three important areas: 1. A blocking zone where the articulated object cannot reach its objective. 2. A "reaction" area where the articulated object changes its behavior to find a new way to the objective. 3. A "out of range" area where the objective is too far from the articulated object and therefore cannot be reached.

Looking at figure 7b we notice that low fitness mean happens over very short time periods showing the ability of the genetic algorithm to quickly converge to optimal or near optimal solutions.

4. Conclusion

In this paper, we have presented the different kinds of interaction with articulated objects that PROVIS features, in order to construct wire networks or to test maintenance schemes. We presented the model we use to achieve interaction with 3D articulated objects. This model, based on genetic algorithms, allows us to take into account every kind of constraints in an homogeneous way. We now work on the co-evolution of multiple articulated objects to achieve cooperative tasks. These objects simulate the interaction of several manipulators to collectively move satellite components.

5. References

[1] Goldberg D. E., "Genetic algorithms in search, optimization and machine learning", Addison-Wesley, Reading 1989.

[2] Badler N. I., Basky B. A., Zeltzer D., "Make them move: Mechanics, control, and animation of articulated figure", Morgan Kaufmann, 1991.

[3] Genrich H. J., Lautenbach K., "System modelling with high-level Petri nets", TCS volume 13, 1981.

[4] Koga Y., Kondo K., Kuffner J., Latombe J. C., "Planning motions with intentions", Proceedings of SIGGRAPH, 1994.

[5] Zhao J., Badler N., "Inverse kinematics positioning using non-linear programming for highly articulated figures", ACM transactions on Graphics, October 1994.

[6] Holland J., "Adaptation in natural and artificial systems", University of Michigan, Press. Ann Harbor, 1975.

Balet O., Torguet P., Gaildrat V. Caubet R., "Autonomous entities in distributed virtual environments", Proceedings of MMM'96, World Scientific, 1996.

Ellis G., "They're Not Making 'Em Like They Used To: Virtual Reality Saves Time and Money in Manufacturing and Construction", Iris Universe, Summer, 1996.

Luga H., Destruel C., Duthen Y., Caubet R., "Adaptive interaction in virtual worlds using evolutionary algorithms", Proceedings of IEEE workshop on Robot and Human communication, Tokyo, 1996.

Sims K., "Evolving virtual creatures", Proceedings of SIGGRAPH, 1994.

Computing the Spanish Medium Electrical Line Maintenance Costs by means of Evolution-Based Learning Processes*

Oscar Cordón[1], Francisco Herrera[1] and Luciano Sánchez[2]

[1] Department of Computer Science and Artificial Intelligence.
University of Granada, 18071 - Granada, Spain.
E-mail: ocordon, herrera@decsai.ugr.es
[2] Department of Computer Science.
University of Oviedo. Oviedo, Spain.
E-mail: luciano@lsi.uniovi.es

Abstract. In this paper, we deal with the problem of computing the maintenance costs of electrical medium line in spanish towns. To do so, we present two Data Analysis tools taking as a base Evolutionary Algorithms, the Interval Genetic Algorithm-Programming method to perform symbolic regression and Genetic Fuzzy Rule-Based Systems to design fuzzy models, and use them to solve the said problem. Results obtained are compared with other kind of techniques: classical regression and neural modeling.

1 Introduction

In Spain, electrical industries do not charge the energy bill directly to the final user, but they share the ownership of an enterprise (called R.E.E., Red Eléctrica Española) which gets all payments and then distributes them according to some complex criteria (amount of power generation of every company, number of customers, etc.)

Recently, some of these companies asked to revise the rules. One of the proposed modifications involved a redistribution of the maintenance costs of the network. Since maintenance costs depend on the total length of electrical line each company owns, and on their kind (high, medium, urban low and rural low voltage) it was necessary to know the exact length of every kind of line each company was maintaining.

To compute the maintenance costs of town medium voltage lines, there is a need to know which would be the total line length if the instalation made would have been the optimal one. Clearly, it is impossible to obtain this value by directly measuring it, since the medium voltage lines existing in a town have been installed incrementally, according to its own electrical needs in each moment.

* This research has been supported by Hidroeléctrica del Cantábrico, under Contract CN-96-055-B1 with the University of Oviedo, and by CYCIT, under Project TIC96-0778 with the University of Granada.

Therefore, we need to solve the problem using other kind of techniques, which are able to relate some characteristics of a certain town with its maintenance cost [16], and the solution obtained has to verify another requeriment. It has not only to be accurate in the problem solving, but able to explain how a specific value is computed for a certain town. That is, *this solution has to be interpretable by human experts*, i.e, to be understood by lawyers and published by the Industry Ministry.

In this contribution, we propose two possible solutions to this distribution problem in the field of Data Analysis (DA). DA can be considered as a process in which starting from some given data sets, information about the respective application is generated. In this sense, DA can be defined as a search for structure in data. Since in our problem there is a need to find a relationship between some variables (the town characteristics and its associated maintenance cost), it is clear that it may be solved by means of DA techniques.

The two said approaches will make use of Evolutionary Algorithms (EAs) [1]. We will consider the use of Genetic Algorithm Program (GA-P) [10] techniques for symbolic regression and the use of Genetic Algorithms (GAs) [8] to design Mamdani-type Fuzzy Rule-Based Systems (FRBSs) [7] to solve the problem.

The paper is set up as follows. In Section 2, we briefly introduce the use of EAs in the field of DA and present the GA-P and Genetic Fuzzy Rule-Based Systems (GFRBSs) [3]. Sections 3 and 4 are devoted to present the two different approaches commented, the use of GA-P algorithms for symbolic regression problems and the use of GAs to design FRBSs. In Section 5, the introduced Electrical Engineering problem is tackled by means of the proposed techniques and their performance is compared with other kind of techniques, clasical regression and neural methods. Finally, some concluding remarks are pointed out.

2 Preliminaries: Evolutionary Algorithms for Data Analysis

Different algorithmic methods for DA have been suggested in the literature, as Clustering algorithms, regression techniques, Neural Netwoks (NNs), FRBSs, EAs, etc. As regards DA in the light of EAs, a representation of the information structure is considered and evolved until having an abstraction and generalization of the problem, reflected in the fitness function.

Recently, a lot of research efforts have been directed towards the combination of different methods for DA. In this way, EAs have been combined with different techniques either to optimize their parameters acting as evolutionary tuning processes or to obtain hybrid DA methods. Next, we briefly introduce two specific hybrid approaches, the GA-P to perform symbolic regressions and GFRBSs. Two particular developments in each field will be presented in Sections 3 and 4.

2.1 GA-P for Symbolic Regression

One of the main applications of the Genetic Programming (GP) [12] is the field of *symbolic regression*. However, the way in which GP perform symbolic regressions is quite restrictive. To solve these GP limitations, a new technique, able to perform symbolic regression by combining the traditional GAs with the GP paradigm to evolve complex mathematical expressions capable of handling numeric and symbolic data, was proposed, the GA-P [10].

The GA-P combines GAs and GP, with each population member consisting of both a string and an expression. The GP part evolves the expression and the GA part concurrently evolves the coefficients used in it. Most of the GA-P's elements are the same as in either of the traditional genetic techniques. A complete description of GA-P can be found in [10].

2.2 Genetic Fuzzy Rule-Based Systems

Nowadays, one of the most important applications of the Fuzzy Set Theory are FRBSs [7]. An FRBS presents two main components: 1) the Inference System, which puts into effect the fuzzy inference process, and 2) the Knowledge Base (KB) representing the known knowledge about the problem being solved, composed of a collection of fuzzy rules. Mamdani-type FRBSs work with the following rule structure:

$$\text{IF } X_1 \text{ is } A_1 \text{ and ... and } X_n \text{ is } A_n \text{ THEN } Y \text{ is } B$$

with $X_1, \ldots, X_n$ and Y being the input and output linguistic variables, respectively, and $A_1, \ldots, A_n$ and B being linguistic labels, each one of them having associated a fuzzy set defining its meaning. The fuzzy rules themselves are collected in a component named Rule Base, while linguistic labels and membership functions are stored in the Data Base (DB).

With the aim of automatically generating the KB, many different approaches have been presented taking EAs, usually GAs, as a base, constituting the so called GFRBSs [3]. These systems have obtained promising results that have extended their use in the last few years (see [5]). For a wider description of GFRBSs and different specific approaches see [3, 4].

3 Interval Valued GA-P for Symbolic Regression

Regression techniques are intended to find an adequate expression for a function g so that given a variable Y that depends on the value of a variable X, $g(X)$ is a good aproximation to Y. In practice, this means that we know N pairs (X_i, Y_i) and we search for a function g such that the mean square error $\frac{1}{N}\sum_{i=1}^{N}[Y_i - g(X_i)]^2$ is minimum [14].

Where the expression of g is not known, the techniques used to solve the problem are known as *symbolic regression.* Note that symbolic regression methods find *punctual* estimates for **Y** given **X**. If we need a confidence interval for **Y** (that is, an interval that contains the unknown value of **Y** with a certain probability) we must resort to fuzzy regression procedures [13] or NN-based ones [11].

We have extended GP methods to the intervalar case [15]. We will search for a set-valued function Γ that depends on m interval parameters and uses interval arithmetic [2] (for example: $\Gamma(x) = k \otimes x$ with $k = [1,2]$ would be $\Gamma(x) = [x, 2x]$. "$\otimes$" means "product" in interval arithmetic) such that the probability that $\mathbf{Y} \in \Gamma(\mathbf{X})$ is higher or equal than a confidence degree β and the mean distance between the extremes of $\Gamma(x)$ is as low as possible.

That set-valued function is defined with a pair of functions g^+ and g^- such that $\Gamma(x) = [g^-(x), g^+(x)]$. The numerical problem that we need to solve is as follows: Given a value ϵ near to zero and *two* independent samples that contain N pairs $(\mathbf{X_i}, \mathbf{Y_i})$ and M pairs $(\mathbf{X'_j}, \mathbf{Y'_j})$ respectively, find two functions $g^-(x)$ and $g^+(x)$ such that

$$\frac{1}{N}\sum_{i=1}^{N}(g^+(\mathbf{X_i}) - \mathrm{g}^-(\mathbf{X_i}))$$

is minimum and the fraction of elements of the *first* sample for which $g^-(\mathbf{X_i}) \leq \mathbf{Y_i} \leq \mathrm{g}^+(\mathbf{X_i})$ is higher than $1 - \epsilon$.

The second sample is used to estimate the confidence degree. β is estimated with the fraction of the M elements of this last sample for which $g^-(\mathbf{X'_i}) \leq \mathbf{Y'_i} \leq \mathrm{g}^+(\mathbf{X'_i})$.

This method can also be used with imprecisely measured data, if the imprecision is characterized by interval values for $\mathbf{X}$ and $\mathbf{Y}$ (for example, data with tolerance: 5±10%).

4 Genetic Algorithms for Learning Mamdani-Type Fuzzy Rule Bases

In this Section, we analyze an specific GFRBS that may be employed as a DA technique. The genetic learning process was first proposed in [4] and it is composed of the following three different stages:

1. An *inductive generation process* for generating Mamdani-type fuzzy rules from examples, with two components: a fuzzy rule generating method based on a non-evolutionary inductive algorithm, and an iterative covering method of the example set.
2. A *genetic multisimplification process* for selecting rules, based on a binary coded GA with a genotypic sharing function and a measure of the FRBS performance. It will remove the redundant rules generated by the previous component with the aim of obtaining different simplified KBs presenting the best possible cooperation among the fuzzy rules composing them.
3. A *genetic tuning process*, based on a real coded GA and a measure of the FRBS performance. It will give the final KB as output by tuning the membership functions in each possible KB derived from the genetic multisimplification process. The most accurate one obtained in this stage will constitute the final output of the whole genetic learning process.

Next subsections will briefly describe each one of the learning stages.

4.1 The inductive generation process

The generation process is based on a previously defined DB, composed of different fuzzy partitions of the variable spaces, as the one shown in Fig. 1.

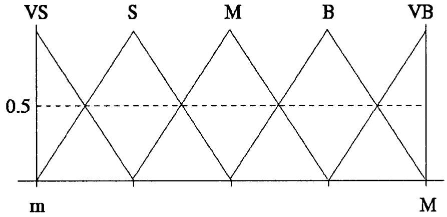


Fig. 1. Graphical representation of a possible fuzzy partition

The covering method is developed as an iterative process that allows us to obtain a set of fuzzy rules covering the example set. In each iteration, it runs the generating method, obtaining the best fuzzy rule according to the current state of the training set, considers the relative covering value this rule provokes over it, and removes from it the examples with a covering value greater than ϵ, provided by the system designer. It ends up when the training set becomes empty.

Each time the generating method is run, it produces a set of candidate fuzzy rules by generating the fuzzy rule best covering every example from the training set. The accuracy of the candidates is measured by using a multicriteria fitness function, composed of three different criteria measuring the covering that each rule provokes over the training set. Their expressions can be found in [4]. Finally, the best fuzzy rule is selected from the set of candidates and given as method output.

4.2 The genetic multisimplification process

Since the generation process works in an iterative way, it may obtain a KB containing redundant rules that do not cooperate adequatelly between them. The aim of this second stage is to simplify the previous KB, removing from it the rules not cooperating well.

The main idea of the genetic multisimplification process, proposed in [6], is that it does not only generate one simplified definition of the previous fuzzy rule set, but several different ones. To do so, it runs the genetic simplification process proposed in [9]. This process is based on a binary-coded GA which encodes the set of rules obtained from the generation process into a chromosome, representing a value 1 the belonging of this rule to the final KB, and a 0 its absence. Two-point crossover and uniform mutation operators are used to alter the individuals and the stochastic universal sampling procedure, along with an elitist selection scheme, to perform selection. The fitness function combines an error measure, the medium square error (SE), and a term penalizing the lack of the encoded KB completeness property.

Each time the genetic simplification process obtains a simplified KB definition, the multisimplification one penalizes the search space zone where it is located to not generate it in future runs. A genotypic sharing scheme is used to penalize individuals according to its space proximity to the previous solutions found. The process ends up when the desired number of simplified KBs is generated.

For a wider description of this process, see [6].

4.3 The genetic tuning process

The genetic tuning process was proposed in [4]. It is based on a real coded GA that encodes a different DB definition in each chromosome. A primary fuzzy partition (as the one shown in Fig. 1) is represented as an array composed by $3 \cdot N$ real values, with N being the number of terms forming the linguistic variable term set. The complete DB for a problem, in which m linguistic variables are involved, is encoded into a fixed length real coded chromosome C_j built by joining the partial representations of each one of the variable fuzzy partitions as is shown in the following:

$$\begin{array}{l} C_{ji} = (a_{i1}, b_{i1}, c_{i1}, \ldots, a_{iN_i}, b_{iN_i}, c_{iN_i}) \ , \\ C_j = C_{j1} \ C_{j2} \ \ldots \ C_{jm} \ . \end{array} \tag{1}$$

The initial fuzzy partitions are used to define the interval of adjustment $[c_h^l, c_h^r]$ associated to every gene c_h in C_j, $h = 1 \ldots \sum_{i=1}^{m} N_i \cdot 3$. Max-min-arithmetical crossover and Michalewicz's non uniform mutation are considered to alter the individuals, and the selection is performed in the same way that in the genetic simplification process. The fitness function is the same used in that process.

On the other hand, the initial DB definition is used to generate the initial gene pool as well. It is encoded directly into a chromosome, denoted as C_1. The remaining individuals are generated in the interval of performance associated to each membership function.

5 Solving the Electrical Engineering Application by Means of the Proposed Evolution-Based Learning Processes

To solve the mentioned electrical problem [16], we were provided with data related to four different characteristics of the towns:

x_1: The sum of the lenghts of all streets in the town
x_2: The total area of the town
x_3: The area that is occupied by buildings
x_4: The energy supply to the town

and to the maintenance costs of line (y) in each one of them in a sample of 1059 simulated towns. Our objective was to relate the last variable (maintenance costs) with the other four ones, first by classical methods and NNs, and later by applying the DA techniques presented in this paper. Numerical results will be compared in the following.

As regards classical methods, we have considered linear, polynomial and NN models. The parameters of the polynomial models were fitted by Levenberg-Marquardt method and the neural model (a three layer perceptron) was trained with the QuickPropagation algorithm. The number of neurons in the hidden layer was chosen to minimize the test error; note that the training error could be made much lower than the shown, but not without making the test error higher. We used 4 input nodes, 5 hidden nodes, and 1 output node.

GA-P algorithms have been applied to check whether we can obtain a formula that is comparable in complexity with the ones considered for the classical solution, while getting better adjust to the real data. We will define "simple expression" as a formula that can be codified in a tree with no more than 50 nodes and depending on no more than 10 parameters. Binary operations will be sum, difference, product, ratio and power. The unary operation will be the square root. We randomly select three individuals every generation. The worst one of them is replaced with the best descendent of the crossover of the remaining ones. Observe that this strategy is elitist and steady state.

Finally, to solve the problem by means of the GFRBS proposed, we have considered an initial DB constituted by some primary equally partitioned fuzzy partitions formed by *five linguistic terms* with triangular-shaped fuzzy sets giving meaning to them (as shown in Figure 1), and the adequate scaling factors to translate the generic universe of discourse into the one associated with each problem variable.

To compare the mentioned techniques, we have divided the sample into two sets comprising 847 and 212 samples, 80 and 20 percent of the whole data set, respectively. SE values over these two sets are labeled *training* and *test.* Results obtained in the different experiments developed are shown in Table 1, where column *complexity* contains the number of parameters and the number of nodes in the parse tree of the expression, as well as the number of rules in the KB of the generated fuzzy model.

In view of them, we can conclude that fuzzy models and GA-P techniques clearly outperform classical non linear regression methods, being equal or superior to NNs. This result has great significance, because it means that NN performance can be achieved with a model with a high descriptive power. Mamdani-type fuzzy models provide the most comprehensive explanation of its functioning, and should be used when a human-readable, rule based, description of the problem is needed. In this case, the GFRBS has found a structure comprising 63 rules. When a mathematical formula is preferred to the rule bank, GA-P methods provide a suitable expression where the user can select the balance between complexity and precision. Punctual GA-P found a mathematical expression that can be codified in 50 nodes (note that a second order polynomial needs 77 nodes and that

Table 1. Results obtained in the problem being solved

Method	Training	Test	Complexity
Linear	164662	36819	17 nodes, 5 par.
2th order polynomial	103032	45332	77 nodes, 15 par.
3 layer perceptron 4-5-1	86469	33105	35 par.
GA-P	18168	21884	50 nodes, 5 par.
Interval GA-P	16263	18325	15 nodes, 4 par.
fuzzy model	19679	22591	63 rules

a linear model uses 17) and that explains the data almost identically than the fuzzy model. Interval GA-P results reflect the ability of the method to eliminate outliers. The results shown in Interval GA-P row were calculated over the subset of examples that were in the confidence interval (97,9% of total) and therefore it found the simplest expression (15 nodes and it does not use the input x_2, the total area of the town). Interval GA-P can only be used when we are allowed to discard some elements of the sample. If we cannot do that, we should apply punctual GA-P or fuzzy models. For example, if we apply the obtained Interval GA-P model to the whole dataset, without discarding outliers, we obtain roughly the same test error, but a much higher training error (202385) that may not be admissible.

6 Concluding Remarks

In this contribution we have solved a real-world spanish Electrical Engineering distribution problem by means of two hybrid EA-based DA methods, the Interval GA-P for symbolic regression and GFRBSs.

Both techniques have demonstrated to be powerful DA tools capable of making abstraction on the data with good generalization properties in view of the results obtained in the application tackled. The first one allows us to obtain expressions with algebraic operators while the second one is able to generate KBs readable by human experts.

References

1. Bäck, T.: Evolutionary Algorithms in Theory and Practice. Oxford University Press (1996).
2. Bojadziev, G.: Fuzzy Sets, Fuzzy Logic, Applications. World Scientific (1995).
3. Cordón, O., Herrera, F.: A General Study on Genetic Fuzzy Systems. In: J. Periaux, G. Winter, M. Galán, P. Cuesta (eds.): Genetic Algorithms in Engineering and Computer Science. John Wiley and Sons (1995) 33-57.
4. Cordón, O., Herrera, F.: A Three-stage Evolutionary Process for Learning Descriptive and Approximative Fuzzy Logic Controller Knowledge Bases from Examples. International Journal of Approximate Reasoning **17(4)** (1997) 369-407.

5. Cordón, O., Herrera, F., Lozano, M.: On the Combination of Fuzzy Logic and Evolutionary Computation: A Short Review and Bibliography. In: W. Pedrycz (ed.): Fuzzy Evolutionary Computation. Kluwer Academic Press (1997) 57-77.
6. Cordón, O., Herrera, F.: Hybridizing Genetic Algorithms with Sharing Scheme and Evolution Strategies for Designing Approximate Fuzzy Rule-based Systems. Technical Report DECSAI-96126, Dept. of Computer Science and A.I. University of Granada. Spain (1997).
7. Driankov, D., Hellendoorn, H., Reinfrank, M.: An Introduction to Fuzzy Control. Springer-Verlag (1993).
8. Goldberg, D.E.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley (1989).
9. Herrera, F., Lozano, M., Verdegay, J.L.: A Learning Process for Fuzzy Control Rules Using Genetic Algorithms. Fuzzy Sets and Systems (1998) to appear.
10. Howard, L., D'Angelo, D.: The GA-P: A Genetic Algorithm and Genetic Programming Hybrid. IEEE Expert (1995) 11-15.
11. Ishibuchi, H., Tanaka, H., Okada, H.: An Architecture of Neural Networks with Interval Weights and its Application to Fuzzy Regression Analysis. Fuzzy Sets and Systems **57** (1993) 27-39.
12. Koza, J.: Genetic Programming. On the Programming of Computers by Means of Natural Selection. The MIT Press (1992).
13. Kacprzyk, J.: Fuzzy Regression Analysis. Omnitech Press, Warsaw (1992).
14. Ljung, L.: System Identification: Theory for the User. Prentice Hall (1987).
15. Sánchez, L.: Interval-valued GA-P Algorithms. Technical Report. Dept. of Computer Sciences. University of Oviedo. Oviedo. Spain (1997).
16. Sánchez, L.: Estudio de la red asturiana de baja tensión rural y urbana. Technical Report. Hidroeléctrica del Cantábrico Research and Development Department (in spanish). Asturias. Spain (1997).

A New Dissimilarity Measure to Improve the GA Performance

G Raghavendra Rao [1] and K Chidananda Gowda[2]

1. Department of Computer Science and Engineering, National Institute of Engineering,
Mysore - 570 008, India,
grrao@nie.ernet.in

2. Department of Computer Science and Engineering, S.J.College of Engineering,
Mysore - 570 006, India,
kcgbra@giasbga.vsnlnet.in

Abstract. The performance of Genetic Algorithms hinges on the choice of "eligible parents" for carrying the genetic information from one generation to the next. Several methods have been proposed to identify the most promising mating partners for the continuation of the progeny .

We propose, in this paper, a measure of dissimilarity between individuals to be considered along with their actual fitnesses. This would help the emergence of more combinations of chromosomes within the population so that at least a few are better. The more is the dissimilarity between the parents, the better are the chances of producing *fit* children. After the problem introduction, we draw an analogy from biology to illustrate that the method should really work, then proceed with the implementation details and discuss the results.

Apparently the philosophy of this paper contradicts some of the views held in connection with niche and speciation, where breeding within the community is preferred. However the issues involved are different and this aspect is dealt with in detail elsewhere in the paper.

1 Introduction

The GA method as originally suggested by Holland[1], drawing inspiration from the process of natural evolution, promises to intuitively drive the search process along the most promising path at the same time not totally discarding the lesser promising ones. However the original algorithm often suffers from one obvious difficulty- the lack of guidelines for the choice the best set of parents for the next generation. Often it is observed that while in the initial stages the solutions tend to surge in an apparently promising direction, they tend to waver during later stages.

Several methods have been suggested to overcome the difficulty in the identification of the most promising set of parents. The concepts of proportional selection, linear ranking, uniform ranking, dynamic or static allocation mechanisms, fitness scaling, tournament selection etc. have been tried with various degrees of success in different problems. While the very nature of GA would limit a universally applicable method of the choice of parents being evolved, we suggest to get back to nature, which in the first instance gave us the concept of Genetic Evolution, for some more hints.

2 Going back to nature

Nature has one curious, but almost universal principle. In all "higher" beings, it tries to discourage incest - the mating of genetically similar individuals. In plants it would use different mechanisms to promote cross pollination bypassing self pollination, in animals inhouse mating is discouraged ,even made dangerous, while genetically distant individuals are normally known to beget better quality offspring. The logic appears to be a simple one - parents with vastly dissimilar gene patterns tend to throw up better gene combinations in their offspring, (not withstanding the equal probability of producing hopelessly bad ones -"lethals") because they can combine in many more ways. Since GA is all about the fit individuals, never mind a few hopelessly bad individuals produced along the way. But what about the "fitness" of the individuals in the traditional GA sense? It should still matter. So, we suggest to concentrate on fit individuals as before but try to cross them with a genetically far off relative avoiding the near ones. Essentially each generation is produced with greater turbulance - so that more patterns appear and by the simple law of averages some of them will do better than their parents.

3 Methodology

The methodology assumes the existence of one important feature in the coding format - each gene of the individual will assume one out of a finite possible values, which is definitely the case with most coding patterns. For example a gene with a 4 bit binary coding pattern can utmost have 16 values and at a given time will have assumed any on of those values in a given individual. (Of course, if some of the values are illegal, the algorithm will take care of them.) Obviously, considering two individuals A and B, if a gene i has the same value in both of them, then A and B are similar with respect to i, otherwise their dissimilarity will be the difference in their values of i. If A and B are each of N genes, we define the Dissimilarity between them as the sum of the differences

$$D_{ab} = \sum_{n} (V_{ia} \sim V_{ib}). \qquad (1)$$

where V_i is the value of the i th gene the respective individual.

(As of now, we are giving similar weights to each of the gene values for the purpose of evaluation of dissimilarity but problem specific weights may produce better results).

The problem of allocation of children to each of the fit individuals as suggested in the Simple GA[1] is as follows- Each parent produces a number of genetic copies identical to itself, the number being determined by the parent's fitness (reproduction).

These strings enter the mating pool, cross among themselves randomly to produce the children of the next generation (crossover).

We suggest that the crossover stage can be made more systematic by introducing the concept of dissimilarity. Accordingly, a string will have more affinity towards a highly dissimilar string in comparison with a more similar one. The introduction of this modified crossover mechanism is a vital modification and is dealt with some detail in the next section.

3.1 The modified cross over scheme: One will encounter a slight difficulty if the dissimilarities are to be evaluated directly. If say P is the population size, then one will have P*(P-1) /2 pairs to reckon with, which is a fairly large number even for medium sized populations. However, we intend to overcome this difficulty by settling for the first K better individuals to carry on the reproduction process. This is not likely to seriously affect the performance, since in many cases, the least fit individuals normally get very few copies as their share and automatically fade out.

Considering a total of K individuals being chosen to produce a population P for the next generation, we generate a (K+1) X K dimensioned Dissimilarity Matrix D. While D(1,k) elements store the number of copies that the individual is entitled to produce, based on it's fitness, D(i ,k) i >1, elements would register the dissimilarity measure between the i-th and the k-th individuals. Also, the individuals appear in the descending order of their fitnesses in the matrix. Starting from i=2, we cross the copies of the i-th individual with the copies of the others in proportion to their dissimilarity. i.e. greater the Dissimilarity between two strings, greater is the probability of their crossing (as compared to the totally random probabilities in Simple GA). The first row elements of the matrix D are suitably updated to reflect the remaining uncrossed elements in the pool. (When the most dissimilar string cannot supply the necessary number of copies, the balance is passed onto the next string pattern in the list).

It may be noted that the fitness of the string is still an important criterion in the crossover selections. We do not simply allow the most dissimilar elements in the pool to cross, but provide the opportunity in the order of their fitnesses. Further the crossover probability between similar elements is only reduced but nót made zero. These steps will ensure that the existing building blocks are not indiscriminately broken down during crossover.

4 Experiments

The methodology was adopted to two popular problems to study the performance. The first was the Travelling Sales Person (TSP) problem and the results were compared to those got by the algorithm suggested by[8]. With the considerations of the length of the paper in mind, we do not intend giving the details of implementation except that we compare the performance of the reference method, which is a modified GA method for TSP with our method which is generalised GA method. The results indicated in the table were quite comparable to those obtained by applying the algorithm of [8] for the same distance matrix and are tabulated in table 1.

The second problem was that of a maximum flow problem as depicted in[7]. A network of branches, each with a maximum allowable flow and nodes whose capacities are unlimited but should handle equal inflow and outflows at steady states (balanced) was considered to maximise the flow through the network. The results as compared to those got by applying the method of [7] appear in Table 2.

Cities	popu size	as per ref value gener	our method value gener
20	50	196 24	198 19
50	100	438 28	415 25
75	100	956 38	897 33
100	125	1014 48	1008 42

TABLE 1. T S P: Results as compared to those got by applying the method of [8] as ref (Gener indicates the generation in which the value was arrived at).

Trial no	as per ref value gener	our method value gener
1	90 42	90 45
2	89 48	90 42
3	90 68	90 46

TABLE 2. MAX FLOW PROBLEM : Results as compared to those got by applying the method of [7]. As ref.

5 Conclusions

The results indicate considerable improvements in terms of the number of iterations over the other methods, as is evident from the tables 1 & 2. Also the complexity of each iteration itself is much less in the suggested method. But we further claim that the major advantage of the method lies in it's simplicity . We do not suggest a total modification of GA, but only the crossover function is being modified. Further, the method is problem independent as it does not draw any problem specific knowledge for its operation.

That the concept works well should not surprise us. It essentially creates more "turbulance " in the genetic pool thus throwing up more configurations than the traditional GA. This of course works well initially. But once most of the building blocks are in place does the turbulance not break them up? Apparently not all of them get broken down since even the near similar ones do cross amongst themselves though with a lesser probability. But more importantly to survive and to make to the next generation, the fitness continues to be an important criterion and hence the near optimal ones always survive. Since, in any case, the best individual so far is always being remembered, the best normally will be the winner.

When tried on multimodal optimisation problems, the scenerio may become different, because the issues involved are different. When we want different niches to be carved out for different species, we resort to a level of inbreeding, thereby separating out the different peaks. Any mating between two highly fit parents from two different niches would produce "lethals" (A similar disaster as would result from a man crossing with a cat[2)]) so mother nature guards against such an eventuality. But it also prevents the pollen of a flower pollinating a flower from the same plant and children of the same parents are not preferred mates. The thumb rule could be wherever a sharing (niche formation for multi modal solutions) is not involved, a cross between dissimilar strings is desirable.

Because of space limitations in our machines for holding large sized populations are concerned, we generated our own TSPs randomly and compared their performance with those obtained from the application of specialised method. The results indicated are the best performance results. However, since we aimed at suggesting a general GA solution rather than a very good TSP or a Maximum flow problem ,the results are highly encouraging.

References

1. J.H.Holland, "Adaptation in Natural and artificial systems" , Ann Arbor , MI, Univ. Michigan Press (1975)
2. D E Goldberg, "Genetic Algorithms in search,optimisation and machine learning ," Reading MA : Addison-Wesley (1989)
3. P Jog , J.Y. Suh and D.V. Gucht, "the effects of population size, heuristic crossover and local improvements in genetic algorithms for the TSP", in Proceedings of 3rd international conference on genetic algorithms, San Mateo, CA: Morgan Kaufmann (1989)
4. D Whitley, T Starkweather and D'Ann Fuquay, "scheduling problems and Travelling sales person : the genetic edge recombination operator" in the proceedings of the 3rd international conference on Genetic Algorithms ,Morgan Kaufman (1989) pp133-140.
5. K.A. Dejong and W.M. spears, "using GA to solve NP-complete Problems ", in the proceedings of the 3rd international conference on Genetic Algorithms, Morgan Kaufman (1989) pp124-132.
6. W.Wienholt, "a refined Genetic Algorithm for parameter optimisation problems", in the proceedings of the 5th international conference on the genetic Algorithms, Morgan Kaufman (1993) pp589-596.

7 T.Munakata and D J Hashier, “A Genetic Algorithm applied to maximum flow problem” in the proceedings of the 5th international conference on genetic algorithms, Morgan Kaufman (1993) pp 488-493.

8. A Homaifer , S Guan and G E Lieping “ A new approach to Travelling Sales Person by Genetic Algorithms” ,in the proceedings of the 5th international conference on Genetic Algorithms ,Morgan Kaufman (1993) pp 460-466.

9. M Srinivas and L M Patnaik “Genetic Algorithms -A survey”, IEEE Computers (June 1994) pp17-26

STRPLAN: A Distributed Planner for Object-Centred Application Domains*

Rafael Berlanga Llavori

Departament d'Informàtica, Universitat Jaume I
Campus Penyeta Roja, E-12071 Castellón
E-mail: berlanga@inf.uji.es

Abstract. This paper presents a novel planner, called STRPLAN, which is intended to generate and validate parallel plans for structured domain models. Structured domains are here defined as a specialization of the Loosely Sort-Abstracted (LSA) models [13], which regards most of the features of object-oriented data models. The operational semantics of STRPLAN are based on the decomposition of incomplete plans into subplans that are solved separately. The criterion to produce this decomposition is given by the concept of domain region, which is here envisaged as a grouping of substate class expressions.

1 Introduction

In the last years, the advent of the so-called software domains has produced a great interest within the AI planning area. Applications such as information gathering in the web, software agents and distributed databases are some examples of these domains. These applications have offered well-known and practical frameworks over which to test and apply current AI planning techniques and algorithms. However, state-of-the-art planners have not been able to handle the big size and complexity of these applications. As a consequence, several ad-hoc algorithms have been developed for them (e.g. [8][5]).

On the other hand, recent research in planning has tackled the inherent complexity of propositional planning by defining new encodings for both domain description models [11] and plan representation [4]. The former trend is mainly intended to change the point of view of knowledge engineers from propositions to objects. Conversely, the latter trend is aimed at obtaining more sophisticated forms of plan representations for classical domain models, and in this way to speed up planning search. From our point of view, both trends can constitute a good starting point for the definition of more powerful planners able to handle large size and complex applications.

This paper describes a novel planner, STRPLAN, which is intended to generate and validate parallel plans for structured domain models. In this paper, structured domain (SD) models are defined as a specialisation of the so-called Loosely Sort-Abstracted

* This work has been partially funded by the Spanish C.I.C.Y.T. project TEL97-1119.

(LSA) models [11], which regard some of the features of object-oriented data models. However, the here proposed domain description language has been specially designed to provide a better support for object-oriented domain models as those used in generic knowledge based systems like [6][12].

Regarding the operational semantics of STRPLAN, it is based on the decomposition of plans into parallel subplans that can be solved and co-ordinated separately. The criterion for achieving this plan decomposition is given by the concept of *domain region* [2][3], which is defined in terms of the substate definitions. The STRPLAN planner takes advantage from this plan representation to define a fully distributed architecture of simpler separate sub-planners. In the future, this architecture will allow very large applications with a great variety of domain regions to be handled by a single planner.

2 Overview of Structured-Domain Models

Broadly speaking, a SD-model is a specialisation of the so-called Loose Sort-Abstracted (LSA) models [11]. Our objective is to design a description language similar to those defined in generic object-oriented knowledge systems like [6][12]. For this reason, the predicates of our description language are limited to express pairs of attribute/value so that they describe object states. This fact reduces the expressiveness of the description language because only binary predicates are allowed. To overcame this limitation, we have introduced into our language both complex sorts (i.e. sets and tuples) and integrity rule specifications. The following paragraphs introduce the different components of our description language as well as some concepts that are necessary to define the consistency of SD-models. Formally, the description language of a SD-model [2][3] consists of:

- A set of class *attribute* names $\mathcal{A}tt$.
- A set of *sorts* divided into: class names $C\mathcal{N}$and type names $\mathcal{TN}$.
- A set of *constants* divided into: object identifiers $OI$and atomic values $\mathcal{V}$.
- A set of binary *predicates* $\mathcal{P}red$.
- A set of *rules* expressing the domain integrity constraint IC.
- A set of well-formed *substate* expressions $\mathcal{E}xpr$.
- A set of well-formed *region* expressions $\mathcal{R}eg$.

Each sort s of the language has associated an extension, denoted $dom(s)$, which is a subset of either OI or $\mathcal{V}$ depending on whether the sort s is a class or a type respectively. Like in LSA models [11], non-primitive sorts (classes or types) can be constructed from others by grouping their extensions. In this case, the non-primitive sort is said to be a *super-sort* of its components. Furthermore, our description language also allows for complex sorts, such as sets of a sort s, denoted with $\{s\}$, and tuples of different sorts $s_1, ..., s_n$, denoted with $[s_1, ..., s_n]$. The extensions of these complex sorts are respectively sets and tuples of constants belonging to their respective sorts.

Every variable in the formalism is associated to some sort s, so that it ranges over its extension, i.e. $dom(s)$. Primitive sort variables will be denoted with the name of the sort and a numeric subscript (e.g. Box_1). Additionally, variables that refer to class sorts will be named *object variables.*

A class C of a SD-model represents the common structure of all the objects belonging to its extension. This structure is described in our formalism by using a set of *structural predicates* of the form:

$$att\_name(Var_1, Var_2)$$

Where $att\_name \in \mathcal{Att}$ represents an object attribute, Var_1 is an object variable of sort C that represents the set of objects of this class, and Var_2 is a sort variable that represents the values that can take the object attribute, that is, its *type*. Furthermore, structural predicates can be classified into *static* and *dynamic*, where static predicates are those that cannot be changed by any action of the domain model.

Example: The following table describes the structural predicates of some E-STRIPS's domain classes [7]:

Structural predicates	*Type*
next(Mv_Object, Object₂)	dynamic
in(Object, Room)	dynamic
near(Object, Door)	dynamic
status(Obejct, St_obj)	dynamic
holds(Agent, Object)	dynamic
status(Door, St_door)	dynamic
opens(Key, Door)	static
connects(Door, [Room₁, Room₂])	static

A substate of an object o is a set of groundings associated to a set of dynamic structural predicates whose first argument unifies with o. In a domain model, substates are intended to describe the parts of object states that are subject to changes. In order to manage all the possible domain substates, the concept of substate expression is introduced in SD models. These expressions are declared as follows:

$$S\_name(V_1,..,V_n) := List\_Pred$$

where S_name is the name of the substate class expression, and $V_1,...,V_n$ are the object variables that are included in the list of predicates $List\_Pred$. V_1 is called the reference object variable. Each grounding $(o,...,v_n)$ of a substate class expression $S(V_1,...,V_n)$ expresses a consistent substate of the reference object o.

2.1 Domain Regions and Actions

In this section, we define the concept of domain *region* and introduce it into the formalism above. We conceive domain regions in a similar way to Lansky [9][10], that is, as a classification of substates and actions in terms of object interactions and agents. This classification is aimed to decomposing goals and plans such that they can be solved separately and then coordinated. This strategy constitutes an effective way to treat multi-agent planning problems [9]. However, on the contrary to Lansky's, our regions are based on sub-states instead of actions, and can be *automatically* inferred from the domain description [2].

In the context of a SD-model, a domain region is defined as a set of substates that share the same reference objects and a common set of structural predicates involving this object. In practice, domain regions are described by means of region expressions as follows:

$$R\_name(O) := List\_substates$$

where *R_name* is the name of the domain region, *O* is the reference object variable, and *List_substates* is the list of substates that are ascribed to the region.

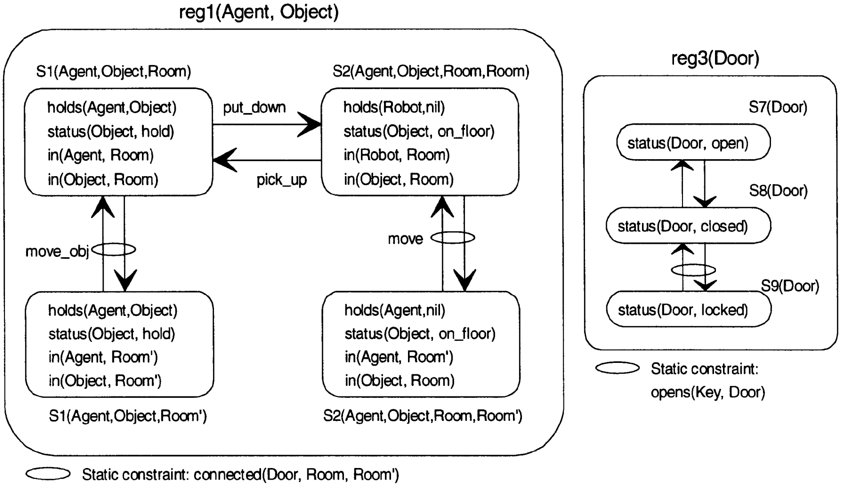


Fig. 1. Example of regions, substate classes and transitions for the E-STRIPS domain.

As mentioned above, regions are intended to classify actions and facilitate the decomposition of goals and plans. Therefore, actions must be defined in terms of the domain regions. In particular, we define an action as a conditioned transition from one state to another within a same domain region, which is called *activity region*. Furthermore, the condition of such a transition must be a set of substates belonging to external regions.

Formally, we define the domain actions as follows:

$$A\_name(V_1,..,V_n) := (Reg, Pre, Post, Cond)$$

where *A_name* is the action schema name, $V_1,...,V_n$ are the sort variables that describe the arguments of the action schema, *Reg* is a region expression that represents the activity region, *Pre* and *Post* are different substate class expressions belonging to the region *Reg*, and *Cond* is a list of substate class expressions belonging to regions distinct of *Reg* (i.e. prevail conditions). The condition of an action can also include static and relational predicates. From now on, we will use the functions *Reg*(*a*), *Pre*(*a*), *Post*(*a*) and *Cond*(*a*) to denote the *a*'s components.

The operational semantics of the domain actions can be briefly outlined as follows. An action *a* is only applicable within domain regions that unify with *Reg*(*a*) and whose substates unify with *Pre*(*a*). The result of applying an action to a domain region is the new substate *Post*(*a*), regarding the set of unification constraints implied by the *a*'s application.

As an example, consider the following definition for the action *move_obj* in the substate class transition diagram of Figure 1:

move_obj(Agent, Object, Room1, Room2, Door) :=
(*reg1(Agent), S1(Agent, Object, Room1)*,
S1(Agent, Object, Room2),
(*S5(Agent, Door), S7(Door), connected(Door,* [*Room1, Room2*]))

where the prevail conditions are the substates *S5*(*Agent, Door*), which belong to the domain region *reg2*(*Agent*), *S7*(*Door*) which belongs to the region *reg3*(*Door*), and the static domain constraint *connected*(*Door,* [*Room1, Room2*]).

2.2 Causal Chains

In order to obtain certain region states, actions must be sequentially composed. At this respect, it is said that the action *a* concatenates with the action *b* if the region substates of *Pre*(*b*) and *Post*(*a*) unify. Sequences of actions concatenated in this way are named causal chains. Like actions, causal chains can be handled through schema expressions as follows:

$$C\_name(V_1,...,V_n) := (Reg, Pre, Post, Actions)$$

where *C_name* is the name of the causal chain, $V_1,...,V_n$ are sort variables, *Reg* is the activity region of the causal chain, *Pre* and *Post* are different substate class expressions belonging to the region *Reg*, and *Actions* is the set of action schemata that conform the causal chain. We will use the functions *Reg*(*ch*), *Pre*(*ch*) and *Post*(*ch*) to denote the components of a causal chain *ch*, the term *Ch*[*i*] to denote the i-th action of the causal chain *Ch*, and *length*(*Ch*) to denote its number of actions.

3 Plan Representation

In previous sections, we have defined substate expressions and actions in terms of domain regions. Furthermore, actions can be sequentially composed and then grouped into causal chains. As a consequence, plans can be defined only in terms of the domain regions and causal chains implied by a SD-model.

Thus, a plan can be defined as a set of parallel subplans, each of them assigned to a different domain region. These subplans are composed of causal chains, which must be sequentially composed since they are acting on a common activity region. One important aspect of maintaining a set of parallel subplans is the coordination between their actions [9]. In our case, this coordination is necessary because action conditions are established in external regions, i.e. they participate in different subplans. For this purpose, we introduce a discrete and linear time model to associate each subplan element to a time constant.

With this approach negative interactions between actions (i.e. conflicts) are managed through the correct composition of causal chains and the coordination of the action external conditions.

Specifically, two kinds of time expressions can be included in the subplans:

- $S{:}t$, which states that S is a substate expression that happens at the time instant t.
- $Ch{:}[t_0...t_f]$, which states that the actions of chain Ch will be sequentially executed at times $t_0...t_f$ respectively.

From the definitions above, we can formally define a plan as follows:

$$Plan := (Plan[Reg_1], ..., Plan[Reg_n])$$

where each $Plan[Reg_i]$, with $1 \le i \le n$, is the subplan associated to the activity region Reg_i and is formed by a totally ordered sequence of time expressions involving substates and causal chains ascribed to Reg_i.

The following definitions formally state what a planning problem is and which is the form of its solutions. As usual, a planning problem is specified with an initial situation SI of the domain and an intended goal situation SG. The solution of a planning problem is a plan that achieves the goal situation when applied to the initial situation.

Definition 1. A plan $P := (P[R_1],...,P[R_n])$ is a solution to the planning problem $\langle SI, SG\rangle$ if and only if:

1. Each subplan $P[R_k]$ $(1 \le k \le n)$ is formed by an alternate sequence of substate expressions and causal chains of the form:

   $$P[R] = (SI[R]{:}\ t_1,\ ch_1{:}[t_{1,1} ... t_{1,s}],\ S_1{:}\ t_1,...,ch_n{:}[t_{n,1} ... t_{n,q}],\ SG[R]{:}t_\infty)$$

   such that $Pre(ch_1) \approx SI[R]$, $Post(ch_n) \approx SG[R]$ and for each ch_i $(1< i < n)$, $Pre(ch_i) \approx S_{i-1}$ and $Post(ch_i) \approx S_i$, where the symbol $\approx$ states for unification.
2. For each time expression $ch{:}[t_{1,1} ... t_{1,s}]$ in a subplan of P, for each i $(1 \le i \le length(ch))$, and for each substate expression $S \in Cond(ch[i])$, there must exists a time expression $S'{:}\ t$ in some subplan of P such that $S \approx S'$ and $t \in \{t_{1,1} ... t_{1,s}\}$.

If some of the conditions above are not satisfied, the plan P is said to be incomplete. Each subplan $P[R_k]$ that does not satisfy the condition (1) is said to be an incomplete subplan.

4 STRPLAN Architecture

STRPLAN differs from most of the current planners in that it relies on a fully distributed architecture. This means that a planning problem is solved by decomposing it into subproblems, which are in turn solved by a set of separate subplanners. The final solution is obtained by combining the local solutions of these planners. The generic architecture is depicted in Fig. 2. Next sections are dedicated to describe in detail each of its components.

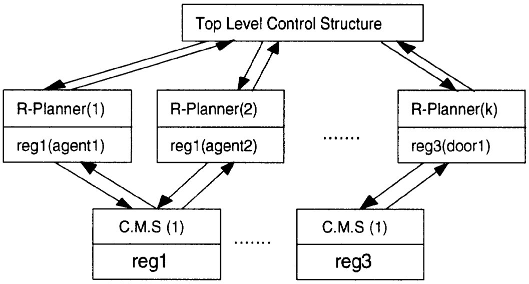


Fig. 2. STRPLAN Planner Distributed Architecture.

4.1 R-planners

A R-planner consists of a domain region R, an input stream that contains the required conditions from external subplans and a set of consistent computation states. A computation state is a pair $\langle Sq, \sigma\rangle$ where Sq is a possibly incomplete plan for the region R and σ is a set of binding constraints over the variables in Sq. A computation state is said to be consistent if all the variables satisfy the constraints in σ.

Broadly speaking, a R-planner is intended to fix one by one those flaws of an incomplete *global* plan that concern with its *local* subplans. According to the plan definition given in Section 3, these flaws can be classified into two categories [2]:

- *Causal Gaps*. A causal gap is produced in a subplan whenever two non-unifiable substate expressions appear contiguously.
- *Open Conditions*. An open condition is an action condition that has not been associated to a substate expression within some subplan.

$S{:}t$ is an open condition that does not interfere with, and is not embedded by any chain in Sq $\langle Sq, \sigma\rangle \rightarrow \langle Sq, \sigma\rangle$ where SQ' is the result of inserting $S{:}t$ into SQ	$S1$ and $S2$ form a causal gap in Sq $\langle Sq, \sigma\rangle \rightarrow \langle Sq', \sigma'\rangle$ where Sq' is the result of inserting a nexus causal chain ch between $S1$ and $S2$ such that $Pre(ch) \approx S1$ and $Post(ch) \approx S2$. σ' is the result of adding the constraints of ch to σ.

Fig. 3. State transitions for an R-planner.

Fig. 3 shows the computation state transitions that an R-planner applies to fix the subplan flaws. Since a causal gap may be fixed by more than one causal chain, the second transition rule is non-deterministic. If there no exists any transition rule to fix an incomplete subplan, the R-planner fails. In this case, the planner backtracks to the choice points left by the second rule. If all the options of a choice point fail, then the R-planner returns to the Top Level Control Module (TLCM) the open condition involved by the failing causal gap and stays in the last consistent computation state.

Additionally, in order to coordinate the separate R-planners, each R-planner must send to the TLCM all the external conditions of the inserted causal chains.

4.2 Chain Management Systems

The synthesis and insertion of causal chains is an essential operation when fixing incomplete subplans (see Fig. 3). A chain management system (CMS) is a module in charge of producing, and eventually storing, the causal chains required by a set of R-planners that share a common region type. In the current approach, all the CMS have the same structure and behaviour.

The CMS mainly relies on a goal regression algorithm to generate causal chains for the input causal gaps. CSP techniques are here used to reduce the domains of the involved object variables. The algorithm returns as result a partially instantiated causal chain plus a set of unification constraints over its object variables.

Fig. 4 depicts an example of causal chain generation. The algorithm input is a causal gap in a region *R*. The initial goal set only comprises the final substate of the causal gap. A new goal set is obtained with the post-conditions of those actions whose pre-conditions are satisfied by the preceding goal set. The algorithm stops either when the current goal set contains some substate that unifies with the initial one, or when no actions can be applied to the current goal set. In the latter case, the causal gap is recorded as a no-good (i.e. causal gaps that have no solutions).

Additionally, the CMS generalises and stores the causal chains generated by the previous algorithm. This process also regards no-goods. It is worth mentioning that the generalisation method applied to causal chains is quite similar to that of macro-actions [11].

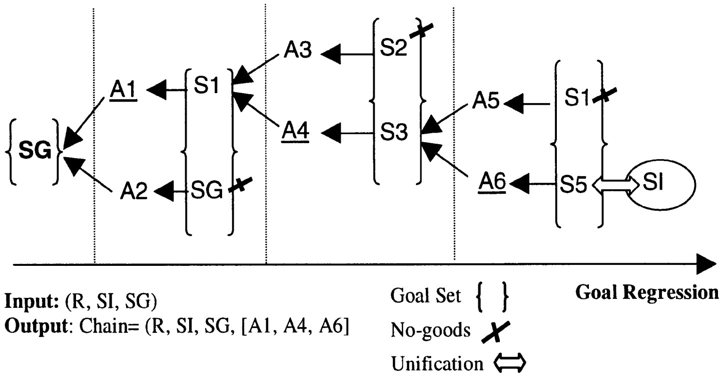


Fig. 4. Example of causal chain generation in the CMS.

4.3 The Top Level Control Module

The Top Level Control Module (TLCM) coordinates and creates all the R-planners that are necessary to solve a planning problem. This coordination is done interchanging the open conditions of the sub-plans they are building. Thus, computation states of the TLCM consist of a set of substate expressions received from different R-planners.

Given a planning problem ⟨*SI*, *SG*⟩, the TLCM initially creates as many R-planners as required by the goal *SG*. Let Reg_1, ..., Reg_k be the regions to which the possible substates of *SG* can be assigned. Then, for each i ($1 \leq i \leq k$) the TLCM creates a R-planner whose domain region is Reg_i and its initial subplan is

$$(SI[Reg_i]\text{: } t_\infty, SG[Reg_i]\text{:} t_1)$$

Here t_1 and t_∞ are the earliest and latest possible time points for the global plan.

Once the TLCM has created the initial R-planners, they start working to achieve their complete consistent sub-plans. As a result, the TCLM can receive substate expressions from these R-planners corresponding with the prevail conditions of the chains they are inserting into their sub-plans. This implies updating the computational state of the TCLM.

Much of the TCLM work is employed in selecting substate expressions from its current computation state and deciding to which R-planner they must be assigned. If there not exists such a R-planner[1], the TCLM must create a new R-planner for this domain region. In this case, given that the substate expression S_{cond}: t_{cond} cannot be assigned to any R-planner, a new R-planner with region R_{cond} and the initial plan ($SI[R_{cond}]$:t_1, S_{cond}: t_{cond}) is created.

Since a substate expression can be assigned to more than one R-planner, as long as a R-planner fails to fix an open condition, the TLCM must backtrack and select the next R-planner to which assign the condition. Finally, the TLCM achieves a global solution when all its R-planners have achieved complete consistent computation states, and the current TLCM's computation state has no open conditions. In this case, the TLCM returns all the subplans.

5 Discussion

The use of regions in AI planning was first introduced in the GEMPLAN planner [9]. GEMPLAN regions are there defined as clusters of actions, which are classified in terms of their constraint-based specification. This concept allows constraint checking to be localised, which is intended to reduce the complexity of big planning problems. Thus, GEMPLAN regions are conceived as a search strategy rather than as an inherent property of the application domains. Moreover, GEMPLAN regions must be carefully defined by the knowledge engineer in an ad-hoc manner. As shown in [9], a bad decision in the declaration of planning regions can imply a poorer performance. Much of these drawbacks stem from the domain representation formalism of GEMPLAN, which ignores domain states and represents actions by means of a first order temporal logic. As a result, domain structural properties cannot be properly stated with this formalism. Unlike GEMPLAN, our concept of region is thoroughly based on the domain structure. Moreover, we have designed a domain representation formalism

[1] This occurs when all the regions of the current R-planners do not unify with the region of the substate expression.

that adopts much of the object-oriented paradigm, which is nowadays the best suited for representing structural properties.

By using regions as an action *interference* criterion, plans can be represented as sets of parallel sub-plans whose interactions are stated through the prevail conditions. As a consequence, we have been able to propose a fully distributed architecture for the STRPLAN planner. This architecture is much simpler than that of GEMPLAN and can be easily implemented upon a distributed environment like a computer network.

On the other hand, it is worth mentioning that the idea of including domain structural properties to improve the planning search is not new in the planning area. Specifically, the hierarchical planners from the PRODIGY family [1][7] include algorithms to extract search control rules from the domain structure. However, they rely on classical planning theories that do not regard action parallelism and domain structure. A preliminary evaluation of STRPLAN [2][3] demonstrates that these planners take little advantage from the domain structure in comparison with STRPLAN.

To sum up, in this paper we have focused on the distributed architecture of STRPLAN and its advantages with respect other planning formalisms. This architecture is based on the theory of structural domain regions [2], which allows planning problems to be automatically decomposed into sub-problems, which in turn can be efficiently solved by a set of separate sub-planners.

References

[1] F.Bacchus and Q.Yang "Downward refinement and the efficiency of hierarchical problem solving" *Artificial Intelligence*, Vol. 71, pp 43-100, 1994.

[2] R.Berlanga "STRPLAN: Multi-agent Planning in Structured Domains" *PhD Thesis*, Department d'Informàtica, Univeristat de València, October 1996.

[3] R.Berlanga "Parallel Planning in Structured Domains" *16th Workshop of the UK planning and scheduling special interest group* (SIG'97), Durham (U.K.), December 1997.

[4] A.Blum and M.Furst "Fast planning through plan-graph analysis" In *Proceedings of IJCAI-95,* pp 1636-1642, Montreal, 1995.

[5] J.M.Bradshaw Ed. "Software Agents" Cambridge Mass. MIT Press, 1997.

[6] M.Jarke et al "Concept Base- A deductive Object Base Manager" *Journal of Intelligent Information Systems*, Vol. 4(2), pp 167-192, 1995.

[7] C.A.Knoblock "Automatically generating abstractions for planning" *Artificial Intelligence,* Vol. 68(1), pp 243-302, 1994.

[8] C.T.Kwok and D.S.Weld "Planning to Gather Information" *In Proceedings of AAAI-96*, 1996.

[9] A.Lansky "Localized Search for Multi-Agent Planning" *In Procceedings of the IJCAI-91*, pp 252-258, 1991.

[10] A.Lansky and L.Getoor "Scoped and Abstraction: Two Criteria for Localized Planning" In *Proceedings of the IJCAI'95*, pp 1612-1618, Montreal, 1995.

[11] T.L.McCluskey and J.M.Porteous "Engineering and Compiling Planning Domain Models to Promote Validity and Efficiency" *Artificial Intelligence*, Vol. 95(1), pp 1-65, 1997.

[12] G.Scheiber, B.Wielinga and J.Breuker "KADS: A principled Approach to Knowledge-Based System Development" *Academic Press*, 1993.

An Intelligent Hybrid System for Knowledge Acquisition

Chien-Chang Hsu[1] and Cheng-Seen Ho[2]

[1]Department of Electronic Engineering, National Taiwan University of Science and Technology,
43 Keelung Road, Section 4, Taipei, TAIWAN 106
cch@ailab2.et.ntust.edu.tw
[2]Department of Electronic Engineering, National Taiwan University of Science and Technology,
43 Keelung Road, Section 4, Taipei, TAIWAN 106
csho@ailab2.et.ntust.edu.tw

Abstract. An intelligent hybrid system is proposed. It includes an adaptive human-machine interface and a hybrid case-based reasoning component for knowledge engineering. The adaptive human-machine interface remembers past question and answer scenarios between the system and the end users. It employs a KA-related commonsense base to help user interaction and a housekeeping engine to search and aggregate the data relevant to those the user answered. It discriminates the professional level of the user by the fuzziness of his answers, and provides different interaction patterns for the user accordingly. The hybrid case-based reasoning component hybridizes case-based reasoning, neural networks, fuzzy theory, induction, and knowledge-based reasoning technology. Hybridizing these techniques together properly enhances the robustness of the system, improves the knowledge engineering process, and promotes the quality of the developed knowledge-based systems.

1. Introduction

Knowledge acquisition (KA) has long been recognized as a very hard problem in knowledge engineering. Many techniques have been proposed, e.g., GAS [3], PROTEGE II [14] and KADS [17] to alleviate it. Most of them are meta-tools. The responsibility of the knowledge engineer is to select appropriate building blocks including problem solving methods, KA methods, knowledge representation methods, and inference methods. It is so easy for a junior knowledge engineer to be trapped in his stereotyped behavior during the selection of the building blocks that he is unable to fully exploit the power of the meta-tools. In general, the quality of the developed KA-tools using these meta-tools is heavily influenced by his experience.

It looks a good idea that a technique that can accumulate knowledge engineering experience is introduced to help the knowledge engineers in using a meta-tool. Case-based reasoning is such a method. It relies on previous experience to solve a problem [7, 10]; it comes up with new solutions by adapting the old solutions that have successfully solved previous similar problems. However, a knowledge engineering scenario usually involves much information. Which of it is significant enough in recognizing the previous cases for use is a big problem. How to maintain a case library so that it can be efficiently and effectively used poses another problem. In this paper, we propose a hybrid case-based system that integrates neural networks, induction, and knowledge-based technology to help learn how to focus on the problem and how to narrow down likely hypotheses in a way similar to the expert.

One more interesting feature in our system. Since the understanding of a given problem may vary between a novice and experienced user, our system provides different interaction patterns for the users with different proficiency background. For instance, when a novice is unable to clearly describe a problem, the system will exploit previous episodes in describing a problem to help him. This is done by properly maintaining transaction records along with important human-machine conversations associated with each transaction. In addition, our system maintains a KA-related commonsense knowledge base both to help clarify potential misunderstanding during human machine interaction and to help reduce system questions.

In short, our system integrates an adaptive intelligent human-machine interface, a hybridized symbolic and sub-symbolic (mainly neural networks) reasoning and learning technique, and a KA-experience-based case library to support the use of meta-tools. This integration invites several advantages. First, with the help of the KA-related commonsense knowledge and the transaction repository, it can adaptively guide knowledge engineers with different background to effectively capture problem features. Second, it retrieves past experiences in much the same way as human beings by utilizing distributed neural networks as the parallel matching mechanism. This improves the efficiency of traditional symbol-based case retrieval. Third, it handles the approximate matching problem by using fuzzy theory, which resembles human's behavior and makes case-retrieval more noise-tolerant. Finally, it produces and learns new cases by carefully examining whether they are subsumed in the case library empowered by the induction technique, which not only reduces the growing speed of the case library but also reduces the frequency of re-training the distributed neural networks.

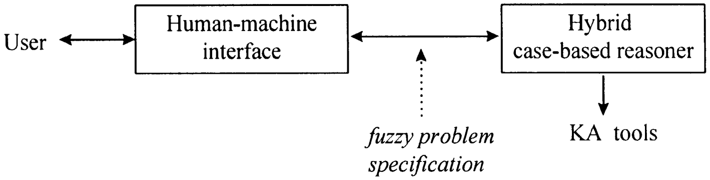


Fig. 1 Architecture of the hybrid intelligent KA-tools-generating system

2. System Architectures

Fig. 1 is the architecture of the hybridized, intelligent system for generating KA tools. It contains two modules, i.e., a human-machine interface and a hybrid case-based reasoner. The former is an adaptive interface employing different levels of interaction patterns to help the users to produce fuzzy problem specification. With the help of the KA-related commonsense knowledge base and the transaction repository, this module also helps less experienced users to produce the specification more efficiently. The hybrid case-based reasoner employs case-based reasoning to use previous KA-tools-generating experience, retrieved against the fuzzy problem specification, to produce a KA tool specification for the current problem. It features the use of a fuzzy neural network to perform parallel matching against previous experiences stored in the case library, the use of induction techniques to derive important features for the given problem from the most similar cases, and the use of rule-based reasoning to perform case adaptation to produce a new KA-tool specification.

3. Intelligent Human-Machine Interface

Fig. 2 elaborates the human-machine interface module. The graphic user interface provides a friendly interface for the user so that he can easily interact with the system. The style of the displayed screen depends on his proficiency level. The specification converter converts the information generated from the knowledge assistant into a fuzzy problem specification and feeds it into the hybrid case-based reasoner to generate a KA-tool generation specification.

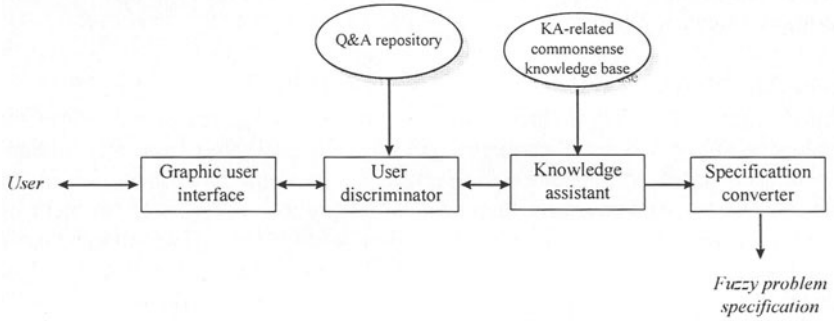


Fig. 2 Intelligent human-machine interface

The user discriminator controls the human-machine interaction. It decides the professional level of the user by measuring the fuzziness of his answer data [5]. The fuzzier the answer, the lower the user level. When the user cannot provide answers, the Q&A repository is invoked to find possible answers. The Q&A repository remembers and organizes all the questions that were previously asked. It serves as a good example of how to fill in missing details for a given question once the question has been answered before. It works with the help of a housekeeping engine, which stores, calculates, selects, and aggregates the data relevant to those the user answers. It conducts statistics on the answered items in the repository and produces proper recommendation to help determine whether to ask further questions or not.

From the set of possible answers returned by the Q&A repository, the discriminator has to choose the best one. It produces two categories of output data. One is the information for controlling the graphic user interface, supplemented with the data from the Q&A repository and the KA-related commonsense. The other contains the answered data from the user, the professional level of the user, the query data, and the checking and verification data.

The knowledge assistant integrates, verifies and validates the problem specification from the user discriminator. It checks whether it is complete, consistent, and relevant to the given environment. It use the KA-related commonsense knowledge base to decide whether the problem specification from the user is reasonable or beyond the capability of the system and/or the meta-tools. The KA-related knowledge base contains the "commonsense about KA", by which we mean the knowledge about interview methodologies [8], meta-tool-dependent heuristics (i.e., selection guidelines), general characteristics of problems and domains, problem assessment methods [16], general user behavior patterns, and cognitive models. We treat it as the basic common knowledge of a knowledge engineer. It does a great help when the system is discriminating user levels; the less experienced user is allowed to skip tricky questions, when the commonsense is capable of taking care of them. The commonsense base also helps the Q&A repository to make the user input more consistent by pruning unreasonable data. If the commonsense detects that the problem might be beyond the capability of the system or not suited to knowledge engineering, it will suggest the user to use other approaches. This self-knowledge capability can dramatically reduce the chance of errors due to inappropriate reasoning. Note that the ontology of this commonsense is quite specific compared with CYC [6].

4. Hybrid Case-Based Reasoner

Fig. 3 shows the architecture of the hybrid case-based reasoner. The case selector takes as input the fuzzy problem specification generated from the human-machine interface and is responsible for selecting the most likely cases stored in the case library. The selected cases are then sent for adaptation to the given problem by the case adapter with the help of the adaptation knowledge. The adapted case becomes the KA-tool specification specific to the given problem. Finally, it will be sent for reuse check to see whether it deserves storage into the case library.

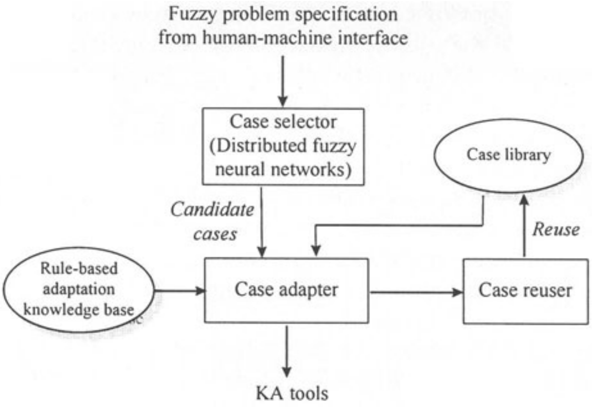


Fig. 3 Hybrid case-based reasoner

The case library contains cases, each representing an instance of the knowledge engineering process. Specifically, a case contains a problem assessment model and a tool generation model [1], corresponding to the problem specification and the way to tackle the problem. The problem assessment model depicts the significant concepts in a problem and the behavior of an existing system that solves the problem [18]. The tool generation model records the scenario of how a KA tool is constructed from a conceptual, "canonical" meta-tool for the problem. This scenario can later be mapped to a specific meta-tool, such as GAS, to actually construct a KA tool. Taking the problem of configuring a VAX 11/780 computer, the target domain of R1 [11], as a case example, Fig. 4 shows part of its case structure.

Problem Assessment Model

• Model specification
⅄ Data model
- Domain: computer systems
- Task: configuring a VAX computer
- Data persistent: sporadic
- Data type: symbolic specification
- Domain terms:
accelerator, adapter, memory, memory controller, CPU, clock battery, disk driver, tape driver, DMA, printer, terminal, power, unibus, supply, repeater, jumper, cabinet, cable, dual port.
⅄ State model
- Goal: computer construction
- Initial state: specification, constraints, configuration
- Snapshot state: selection, constraint satisfaction
- Relationship: (components, constraints), (constraints, configuration)
• Process specification
⅄ Behavior: select, assign, fix, adjust, check
⅄ constraints: select(component, backplane), assign(backplane, box), selection(module, optimal sequence), verify(module, constraint), config(bus, component), assign(box, component), . . . etc..

Tool Generation Model

⅄ Task type: design
⅄ Behavior: propose, specify, configure, refine
⅄ Primitive task: check, propose, construct, recognize, fix
⅄ knowledge model: hierarchical
⅄ knowledge representation: rule
⅄ relationships: (check, propose), (construct, recognize), (recognize, fix/adjust), (fix/adjust, check)

Fig. 4 Partial case structure of configuring a VAX computer

The case selector is a distributed fuzzy neural network containing two layers of nets (Fig. 5). The first layer is the process net, which determines the similarity of the process specification between the input problem and the cases in the case library. The second layer is the model net, which does the similar job on the model specification. Each layer is further partitioned into sub-nets in accord with the task types of the cases [1, 2, 13], namely the analysis type, synthesis type, and hybrid type [9]. The main advantage of this sub-net design is the alleviation of the retraining problem; when a new case is introduced into the case library, we only need to retrain the corresponding sub-nets while keeping the others intact. Moreover, introducing fuzzy representation avoids the problem of matching nothing from the case library.

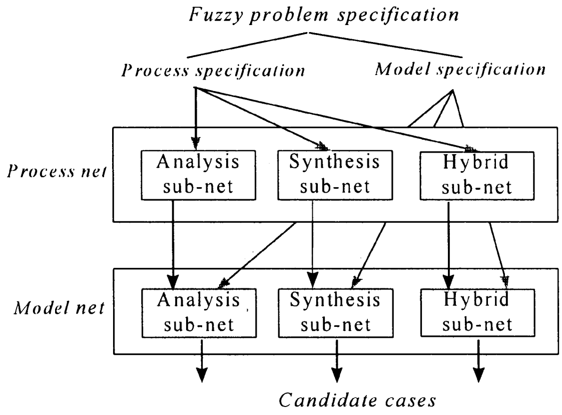


Fig. 5 Distributed fuzzy neural network

The output of the distributed fuzzy neural network contains more than one candidate case. The case adapter is responsible for selecting the best-matched information from all candidate cases and adapting it to solve the given problem. First, we conduct induction on the candidate cases to derive the most appropriate features for the given problem. This is done by summarizing the attribute values from all the candidate cases and using them to construct a constrained induction tree. The basic tree construction algorithm is the CLS algorithm [15]. We use the heuristic that chooses the more commonly-used features for the nodes that are nearer to the root of the tree; that is, a node will have less branches if it is nearer to the root. This design renders the adaptation process easier. Fig. 7 is an example of the constrained induction tree using R1 and REDESIGN [12] as two candidate cases (Fig. 6). The case adapter then uses the adaptation knowledge base to construct a path from the tree as the solution for the given problem.

The case reuser maintains the case library whenever a new case is found. It uses the constrained induction tree to check whether there already exist analogous or subsumed cases in the case library. This approach of case learning features the comparison of all existing candidate cases (represented by the induction tree) at the same time, which reduces the growing speed of the case library and the retraining needs of the distributed neural network.

5. Conclusions and Discussions

We have described an intelligent hybrid system that is self-knowledgeable and capable of performing efficient and effective hybrid case-based reasoning for knowledge acquisition. The former capability is provided by a human-machine interface that remembers the past question-and-answer scenarios between the system and the users. It is able to discriminate the proficiency level of the user according to the fuzziness of his answers and to judge whether a given problem is suitable for knowledge engineering. It also employs a KA-related commonsense base to help user interaction. The user needs not to know the answers of all of the questions. The system also contains a hybrid case-based reasoner that hybridizes case-based reasoning, neural networks, fuzzy theory, induction, and knowledge-based technology to facilitate knowledge engineering. It accumulates experiences as cases in the case library and solves new problems by adapting the old cases that have successfully solved previous similar problems. It employs distributed fuzzy neural networks to perform parallel matching to speed up case retrieval. It uses the fuzzy technique to perform approximate matching to tolerate potential noise in case retrieval. It introduces induction techniques coupled with the knowledge-based technique for case adaptation and reuse. The integration of these techniques quite improves the knowledge engineering process. We are planning to extend the system into a full-fledged hybrid environment for each phase of the knowledge engineering to further improve the quality of the developed knowledge-based systems.

Features	Case #1	Case #2
Domain (D)	Computers	Electronic
Task (T)	Configuring VAX computer	Redesigning TTL circuit
Goal (G)	Computer configuration	Circuit redesign
Initial State 1 (IS1)	Specification/ Constraints	Specification/ Constraints
Process Behavior 1 (PB1)	Check	Focus
Process Behavior 2 (PB2)	Select	Select
Task type (TT)	Design	Design
Tool behavior 1 (TB1)	Propose	Propose
Tool behavior 2 (TB2)	Specify	Check
Constraint 1 (C1)	(AND(PB11, PB12, IS11) (TB11))	(AND(PB21, PB22, IS21,) (TB21))
Constraint 2 (C2)	(AND(TB12) (PB12, TT))	(AND(TB21) (IS1, TT))

Fig. 6 Example of candidate cases

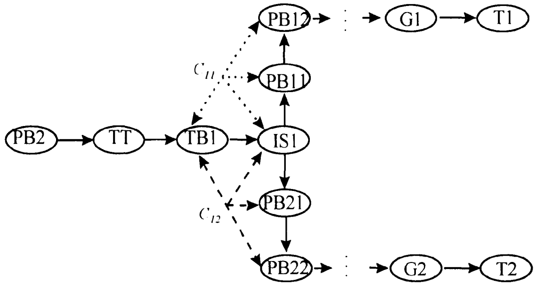


Fig. 7 Example of constructing constraint induction tree

References

1. Breuker, J., Van de Velde, W., (Eds.): CommondKADS Library for Expertise Modeling. IOS Press, Amsterdam Netherlands (1992)
2. Chandrasekaran, B., Johnson, T. R., Smith, J. W.: Task-Structure Analysis for Knowledge Modeling. Communications of the ACM 35:9 (1992) 124-137
3. Chien, C. C., Ho, C. S.: A Primitive-Based Generic Approach to Knowledge Acquisition. Knowledge Acquisition 6:3 (1994) 215-242
4. Chien, C. C., Ho, C. S.: mwKAT: A Meta-Tool-Generated Knowledge Acquisition Tool for Multimedia Workstations Development. Applied Intelligence 5:1 (1995) 51-72
5. George, J. K., Bo, Y.: Fuzzy Sets and Fuzzy Logic Theory and Application. Prentice-Hall, NJ (1995)
6. Guha, R. V., Lenat, D. B.: CYC: A Mid-Term Report. AI Magazine 11:3 (1991) 32-59
7. Hunt, J., Miles, R.: Hybrid Case-Based Reasoning. The Knowledge Engineering Review 9:4 (1994) 383-397
8. Karen, L. G., Karan, H. B.: Knowledge Acquisition. Prentice-Hall, Englewood Cliffs NJ (1989)
9. Kingston, J.: Applying KADS to KADS: Knowledge-Based Guidance for Knowledge Engineering. Expert Systems 12:1 (1995) 15-26
10. Kolodner, J., Mark, W.: Case-Based Reasoning. IEEE Expert 7:5 (1992) 5-6
11. McDermott, J.: R1: A Rule-Based Configurer of Computer Systems. Artificial Intelligence 7:1 (1982) 39-89
12. Mitchell, T. M., Steinberg, L. I., Kedar-Cebelli, S., Kelly, V. E., Shulman, J., Weinrich., T.: An Intelligent Aid for Circuit Redesign. AAAI (1983) 274-278
13. Mizoguchi, R., Tuerino, Y., Ikeda, M.: Task Analysis Interview Based on Task Ontology. Expert Systems with Applications 9:1 (1995) 15-25
14. Puerta, A. R., Egar, J. W., Tu, A. W., Musen, M. A.: A Multiple-Method Knowledge-Acquisition Shell for the Automatic Generation of Knowledge-Acquisition Tools. Knowledge Acquisition 4:2 (1992) 171-196
15. Quinlan, J. R.: Introduction of Decision Trees," Machine Learning I (1986) 81-106
16. Slagel, J., Wick, M.: A Method for Evaluating Candidate Expert Systems Application. AI Magazine 9:4 (1988) pp. 45-53
17. Schreiber, G., Wielinga, B., Brenker, J., KADS: A Principled Approach to Knowledge-Based System Development. Academic Press, San Diego CA (1993)
18. Yen, J., Lee, J.: A Task-Based Methodology for Specifying Expert Systems. IEEE Expert 8:1 (1993) 8-15

Using Neural Nets to Learn Weights of Rules for Compositional Expert Systems[*]

Petr Berka[†], Marek Sláma[‡]

[†] Laboratory of Intelligent Systems, Prague University of Economics,
Ekonomická 957, 148 00 Prague, Czech Republic
phone: (+420 2) 24095465,
fax: (+420 2) 7934749
e-mail: berka@vse.cz

[‡] Institute of Computer Science, Academy of Sciences,
Pod Vodárenskou věží 2, 182 07 Prague 8, Czech Republic
e-mail: marek@uivt.cas.cz

Abstract. Knowledge base for a compositional expert system consists of a set of IF THEN rules with uncertainties expressed as weights. During consultation for a particular case, all aplicable rules are combined and weigths of goals (final diagnoses or recommendations) are computed. The main problem when eliciting such knowledge base from an expert is the question of "correct" weights of rules.

Our idea is, to combine the structure of knowledge obtained from expert with weights learned from data. We choose the topology and initial settings of the neural net (number of neurons, prohibited links) according to the rules obtained from expert. Then, after learning such network, we try to interpret the weights of connections as uncertainty of the original rules.

The paper shows some experimental results of this approach on a knowledge base for credit risk assessment.

Keywords: Compositional expert systems, rule extraction, neural networks, credit risk assessment

[*] Partially supported by the grant MSMT no. VS96008.

1 Compositional expert systems vs. neural nets

A knowledge base of a compositional rule based diagnostic expert system (such as MYCIN [12] or PROSPECTOR [3]) consists of a set of rules in the form

$$\textit{condition} \Longrightarrow \textit{conclusion} \ (\textit{weight})$$

where

condition is a conjunction of literals (propositions or its negations),
conclusion is a single proposition,
weight from the interval $[-1, 1]$ expresses the uncertainty of the rule.

The rules create so called *inference net* where we can distinguish (according to their position in the net) three types of propositions; *questions* (propositions which appear only in conditions of rules), *goals* (propositions which appear only in conclusions of rules), and *intermediate propositions* (propositions which appear both in conditions and conclusions of different rules).

During a consultation, all relevant rules are evaluated (usually using backward chaining from goals to questions) by combining their weights with the weights of conditions. Weights of questions are obtained from user, weights of all other propositions are computed by the inference mechanism. According to Hájek's algebraic theory [4], four combining functions are used during this process:

1. NEG - to compute the weight of negation of a proposition,
2. CONJ - to compute the weight of conjunction of literals,
3. CTR - to compute the contribution of the rule to the weight of the conclusion (this is computed from the weight of the rule (w) and the weight of the condition (a)),
4. GLOB - to compose contributions of more rules with the same conclusion $(w'_1, ..., w'_r)$.

Whereas artificial neural nets are based on complete different paradigm (analogy with biological neurons, distribution of "knowledge" in the whole net, local computations instead of global processor), we can observe surprising similarity between consultation with expert system and classification with trained neural net.

If we analyse behaviour of a single neuron of the form shown in Figure 1, we can describe it as inference in a set of rules of the form:

```
     P1 --> a ( w1)
non(P1) --> a (-w1)
          . . .
     PR --> a ( wR)
non(PR) --> a (-wR)
   True --> a (-b)
```

The weighted inputs of a neuron are realised (computed) using CTR from rules with single literal on the left-hand side. The sum of the inputs and the

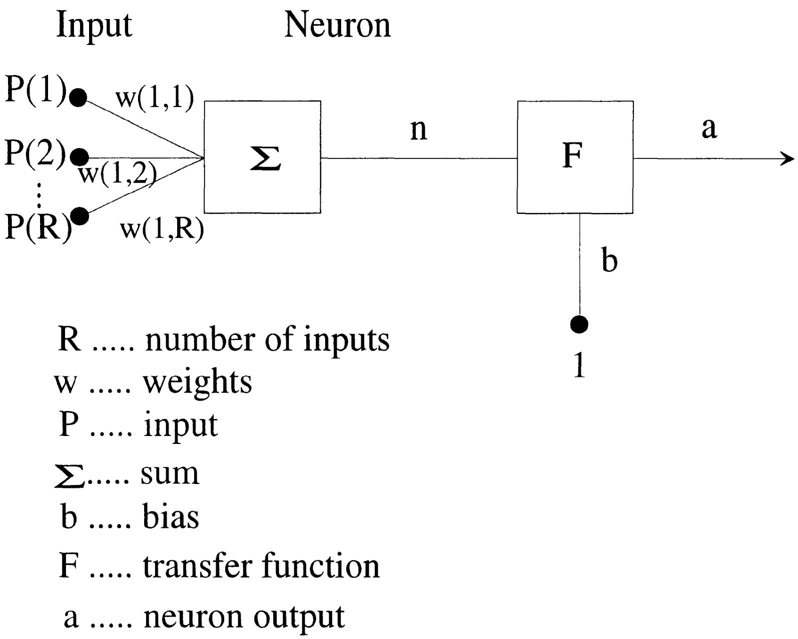


Fig. 1. Scheme of single multiinput neuron

nonlinear transformation of the result are realised with GLOB. Both functions must of course have same effects as the computations in the neuron.

So if the neural net has similar structure as the inference net of rules, we can transform the weights of connections between neurons into weights of rules. Some additional assumptions about the neural net must be done but this is the basic idea behind our approach to knowledge base refinement using artificial neural nets.

2 Case problem - credit risk assessment

We describe our approach on an example of a small, simple, but real knowledge bases which are a part of a greater system for credit risk asessment we are developing together with a big Czech commercial bank [8].

The client (particular enterprise asking for credit) is evaluated from different points of view (eg. incomes, assets, liabilities, etc). These (mainly financial) characteristics of client are grouped into four blocks. At first, the client is evaluated within each of these blocks separately by classifying into one of five categories (excellent, very good, good, poor, very poor). Then, the final rating of the client is given in the same five categories. The knowledge available for classifying clients within each block is in the form of reference values for "excellent" client (e.g. as desirable ranges of values of input attributes [2]).

2.1 Obtaining rules from expert

When we started to automatize this decision process, we created for each block one set of propositions with one goal (`evaluation-is-excellent`) and as many input propositions as is the number of possible intervals for input attributes in given block. We encoded each piece of knowledge (e.g. each criterion of "excellent" client) in the form of complementary rules:

```
IF value-is-OK THEN evaluation-is-excellent WITH WEIGHT (w)
IF value-not-OK THEN evaluation-is-excellent WITH WEIGHT (-w)
```

So, if value of an attribute is OK, we will obtain positive contribution to the final evaluation *excellent*, if value is not OK we will obtain negative contribution. The resulting weight of the goal `evaluation-is-excellent` is transformed into category of the client. This is done by changing the weight (values from interval $[-1, 1]$) into score (values from interval $[0, 10]$).

As an expert system we used the expert system shell SAK. This system offers two sets of combining functions, the *standard* set based on PROSPECTOR-like and MYCIN-like approach and the *logical* set based on multivalued logic [1]. The sets differ in the functions CTR and GLOB.

The main problem when creating the knowledge base by interviewing the expert was how to assign weight to the rules. The expert was able to give us only the structural knowledge, i.e. which input attributes are relevant for particular block and which of them are related together. So we decided to assign equal weights (`w`) to each input attribute within one block, such that if the client fulfils all requirements, his evaluation will be "excellent" and if the client fulfils no requirements, his evaluation will be "very poor".

For each block, we created two different knowledge bases. These knowledge bases differ in how we use additional knowledge about relations between input attributes. If e.g. (according to expert) attributes `B` and `C` are related together, the first set of knowledge bases will contain intermediate proposition `prop-X` [3]. Corresponding rules (all with single literal in conditional part) are shown in Figure 2. In the second set of knolwedge bases we expressed relations between input attributes as conjunctions, so corresponding rules are shown in Figure 3.

We measure the accuracy of our system by comparing (in each block separately) the decision of system with the decision of expert. Table 1. bellow sumarises the results. Column "perfect match" denotes the situation, where both system and expert classified the client into same category, "imperfect match"

[2] The number of input attributes within block varied from 5 to 15.

[3] Since there is only one rule `IF prop-X THEN evaluation-is-excellent WITH WEIGHT (v)` for given intermediate proposition `prop-X`, this proposition is used mainly for explanation.

```
IF B-is-OK   THEN  prop-X   WITH WEIGHT (w)
IF C-is-OK   THEN  prop-X   WITH WEIGHT (w)
IF B-not-OK  THEN  prop-X   WITH WEIGHT (-w)
IF C-not-OK  THEN  prop-X   WITH WEIGHT (-w)
```

Fig. 2. Rules with single conditions

```
IF B-is-OK and C-is-OK  THEN  evaluation-is-excellent   WITH WEIGHT (2w)
IF B-not-OK             THEN  evaluation-is-excellent   WITH WEIGHT (-w)
IF C-not-OK             THEN  evaluation-is-excellent   WITH WEIGHT (-w)
```

Fig. 3. Rules with conjunctive conditions

denotes the situation, where the decision of system differs from the decision of expert in at most one category.

The first set of knowledge bases overestimated the reliability of clients in two of blocks, whereas the second set of knowledge bases tends to underestimate the reliability of clients in all blocks. This is because if conjunction appears in conditional part of a rule, this rule will be activated only if all literals in the conjunction are (with respect to their weights) true. So if e.g. only value of attribute B is OK, in the second knowledge base, the conjunctive rule (with positive contribution) will not fire, whereas in the first knowledge base, we will obtain positive contribution. Nevertheless, the expert prefered the second set, because in this case the system minimized the error of loosing money by granting credit to an unreliable client.

block	first set		second set	
	perfect match	imperfect match	perfect match	imperfect match
1	56%	81%	53%	78%
2	41%	75%	31%	78%
3	19%	84%	47%	72%
4	28%	86%	28%	83%

Table 1. Results for original knowledge bases

2.2 Learning weights using neural networks

After collecting some data, we tried to use neural nets to learn the (missing) weights of our rules. Our expectation was that

1. by tuning weights the classification accuracy of the system will increase,
2. we will find one or two most important attributes within each block, which will help us to formulate some "quicktest" for preclassifying clients as "obiously bad" and "clients worth further contact".

We constructed neural nets for both sets of knolwedge bases. We configure the nets according to the rules obtained form expert e.g. each network has one neuron in the output layer and as many neurons in the input layer as is the number of input attributes in given block. For the first set (where only single literals in conditional part of rules are used), relations between input attributes are expressed as hidden (AND) neurons (see left part of Figure. 4).

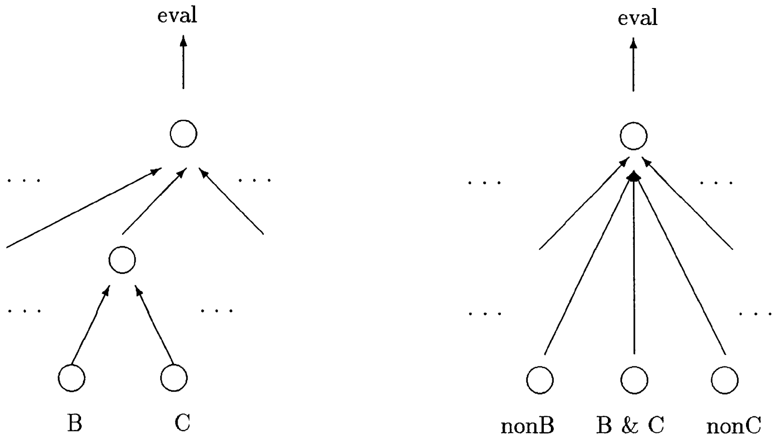


Fig. 4. Topology of used neural networks

The input data were changed into `+1` (if value of attribute was within "correct" range; this corresponds to proposition `value-is-OK`) and `-1` (if value of attribute was out of "correct" range; this corresponds to proposition `value-not-OK`).

For the second set (where relations between input attributes are expressed as conjunctions of corresponding literals in conditional part of rules) additional inputs were created to simulate the evaluation of conjunction in compositional expert systems. So to express rules shown in Figure 3, we compute new input values[4]

$$\begin{aligned}&\text{nonB} = \max(0,-(\text{B-is-OK}))\\&\text{B \& C} = \max(0, \min(\text{B-is-OK},\text{C-is-OK}))\\&\text{nonC} = \max(0,-(\text{C-is-OK}))\end{aligned}$$

In this case, only single layer network was used for training (see right part of Figure. 4). This topology does not benefit from the power of nonlinear computations of multilayer networks, so the performance of the knowledge bases refined

[4] Again, the computation is done for values B and C {-1,1}, the values of new inputs are {0,-1}. The formulae correspond to the function NEG (computed as minus) and CON (computed as minimum) of our inference mechanism.

using this topology is not as high as the performance of knowledge bases refined using network with hidden layers.

The multilayer net was trained by backpropagation algorithm with adaptive learning rate and momentum term. As a transfer function we select the hypertangent sigmoid because the output of this function is in the same range as is used in our expert system for uncertainty processing. We also set the threshold of all neurons to 0. For the single layer net, standard optimalization algorithm was used. The training set consists of about 30 objects, records of clinets evaluated by the expert.

After the training is finished[5] we interpret the weights in neural net as weights of corresponding rules. If sum of weights on input to one neuron exceeds 1 we normalize this weights[6]. We also slightly modified the combination functions of our expert system to propagate uncertainty in the similar way as neural nets:

$$CTR(a, w) = a * w$$
$$GLOB(w_1, ..., w_R) = max(-1, min(1, \sum_w w))$$

The results of testing of the refined knowledge bases are summarized bellow, in Table 2.

block	first set		second set	
	perfect match	imperfect match	perfect match	imperfect match
1	34%	97%	53%	84%
2	31%	94%	41%	84%
3	56%	94%	47%	91%
4	34%	97%	28%	90%

Table 2. Results for refined knowledge bases

So after using neural networks to learn weights, the performance of our rule-based system significantly increases. Our second expectation was fulfiled only partially, only for two blocks we could identify some attributes for the quicktest. Moreower, sometime rules for "correct" values got assigned negative weights (during consultation these rules will give negative contribution to the goal `evaluation-is-excellent`). So the correspondence between neural net and rules is not so straightforward.

3 Related work

Combining symbolic knowledge representation methods and neural networks can be done in basically two ways. One way is to use symbolic knowledge for initial

[5] The performance of the reduced net was lower in comparison to fully connected net with same number of neurons, but our task is to interpret weights not to build a neural classificator.

[6] Inference mechanism of our expert system works with weights of propositions and rules from interval $[-1, 1]$.

setting (topology of net, initial weights, pruned links) of neural networks [11, 5]. Such use of expert knowledge can improve the learning process of neural networks, which remain the main problem solving method in given application.

Another way (more relevant to our approach) is to extract symbolic knowledge from neural network. In this case the extracted knowledge is represented in various forms of IF THEN rules. So Shavlik [11] extracts NofM rules from trained net (initial setting of which is based on expert's knowledge). Kane et all. [6] constrained themselves to neural network with two-valued inputs ($[-1, 1]$) and three-valued weights ($\{-1, 0, 1\}$). From such network they extract logical formulae in disjunctive form. Sestito and Dillon [9] extract rules from fully connected net, so no additional knowledge about the structure of the network is used. By adding outputs as additional inputs they looked for similar weights of links between original inputs resp. "added" inputs and hidden neurons. In case of similarity, they create rule which links inputs with corresponding output. Similarly, Setiono and Liu start their rule extraction from fully connected three layer network [10]. After finding important connections (those with high weights) they prooned unimportant ones and then create rules by discretizing hidden unit activation values. Pedrycz and Roventa [7] introduce the notion of "logical neuron". They use AND neurons to build conditional parts of rules (in the hidden layer) and OR neurons (in the output layer) to agregate rules with the same conlusion. In case of fully connected network, the result of learning are induced rules, in case of network pruned according to expert knowledge (rules), the learning leads towards calibration of these rules.

With exception of the last work, all other methods extract rules which express the IF THEN relations between conditions and conclusions without any uncertainty. So this knowledge is not well suitable for direct use in compositional expert systems. Pedrycz and Roventa use different approach to experessing uncertainty.

4 Conclusions

The task of credit risk evaluation is one of the most popular ones using both expert systems and artificial neural networks. The described approach can benefit both from knowledge based systems (from the structure of knowledge given by human expert) and from neural nets (from their learning ability). To combine such different approaches it is necessary to keep the knowledge representation and processing as consistent as possible. This was achieved by pruning the neural net and fixing some of their parameters and by modifying the inference mechanism of an expert system.

By using machine learning methodology (neural nets) to refine knolwedge base of a "classical" expert system we tried to overcome the well known knowledge acquisition bottleneck. Nevertheless, in this particular application of credit risk assessment, we encountered some problems which can be viewed as "data acquisition bottleneck". Therefore, the testing of our system was not yet performed as thoroughly as we hoped.

To decide, whether the proposed approach can be used more generally, more experiments must be done. The preliminary results make us optimistic.

5 Acknowledgments

We thank the Investment and Post Bank, Inc., Prague and H.E.M Informatics, Ltd., Prague for their support of this research.

References

1. Berka,P. - Ferjenčík,J. - Ivánek,J.: Expert system shell SAK based on complete many-valued logic and its application in territorial planning. In: (Novák, et all.) Fuzzy Approach to Reasoning and Decision Making. Academia, Prague and Kluwer, Dodrecht 1992, 67-74.
2. Berka,P. - Sláma,M. - Beran,H.: Knowledge Base Refinement Using Machine Learning. Neural Network World, Special Issue on PASE'95, Vol 5, No 4, 1995, 393-400.
3. Duda,R.O. - Gasching,J.E.: Model Design in the Prospector Consultant System for Mineral Exploration. In: Michie, D. (ed.), Expert Systems in the Micro Electronic Age, Edinburgh University Press, UK, 1979.
4. Hájek,P.: Combining Functions for Certainty Degrees in Consulting Systems. Int.J. Man-Machine Studies, 22, 1985, 59-76.
5. Ivanova,I. - Kubat,M.: Initialization of Neural Networks by Means of Decision Trees. Knowledge-Based Systems, Vol 8 No 6, 1995, 333-344.
6. Kane,R. - Tchoumatchenko,I. - Milgram,M.: Extraction of Knowledge from Data Using Constrained Neural Networks. In: Proc. ECML'93, Springer, 1993, 420-425.
7. Pedrycz,W. - Roventa,E.: Fuzzy Neurocomputations in Designing Expert Systems.
8. Sekerka,B. - Knaifl,O. - Beran,H. - Berka,P.: Application of AI Techniques in Credit Risk Assessment. Neural Network World, Special Issue on NEUROWORKSHOP'95, Vol 5, No 6, 1995, 1045-1048.
9. Sestito,S. - Dillon,T.: Knowledge Acquisition of Conjunctive Rules Using Multilayered Neural Networks. Int.J. Intelligent Systems, Vol 8, 1993, 779-805.
10. Setiono,R. - Liu,H.: Understanding Neural Networks Via Rule Extraction. In: Proc. IJCAI'95, 1995, 480-485.
11. Shavlik,J.: A Framework for Combining Symbolic and Neural Learning. Tech.Rep. 1123, Uni. of Wisconsin, 1992.
12. van Melle,W. et al.: The Emycin Manual. Stanford University, Dept. Comp. Sci., Report No. STAN-CS-81-885, October 1981.

Case- , Knowledge-, and Optimization- Based Hybrid Approach in AI

Vladimir I. Donskoy

Department of Mathematics, Simferopol State University
4, Yaltinstkaya Street, 333036, Ukraine

Abstract. The Hybrid approach described in this paper presents a powerful tool for development DSS, ES, as well as Data Mining, Pattern Recognition Systems and other AI applications. It's more important advantage is an ability to realize an optimal choice from the set of the admissible alternatives based on the canonical optimization model which can be synthesized by using such AI methods as case-based, rule-based and analogy-based reasoning. Some applications dedicated to the portfolio selection process, regional ecological-economic control, and Dual Expert Systems are described.

1 Introduction

Hybrid approaches in AI have great advantages and perspectives for ES and DSS development. Human Brain's cognitative organization based on the left and right semispheres interactions is an inspirational example of Hybrid natural system in spite of it is very complicated and weakly studied. The following advantages of Hybrid models in AI can be indicated as very important:

- an error probability of decision making decrease;
- possibility of various types of initial information utilization (such as cases, rules, mathematical and other properties of considering objects, and relations between these objects);
- Data and Knowledge verification (for example, cases are used to test a rules system, and rules system is used to test an appearance of artifacts);
- more complete description of the investigating domain;
- possibility to realize an optimization based decision making unlike a dichotomous choice;
- possibility to explain some singularities of data and rules sets.

There are many paradigms for machine learning, associated with case-based learning, rule induction, neural networks, genetic algorithms, analogy, and others [1]. The methods based on these paradigms can be classified on the following classes:

- empirical induction based methods;
- deduction based methods;
- analogy based methods.

Genetic approach can be considered as a special tool for the dynamic processes realization in AI systems. This approach one's self uses induction, deduction, and analogy in a special form, as well as optimization has a role of the tool for the best ways of realization finding when induction, deduction, and/or analogy are used [2]. So case based approach must be considered as belonging to the empirical induction (generalization) based class, and analytic learning (for instance using production as in the Prolog language) - belonging to the deduction based class.

The differences between various realizations of above methods inside of any class play such role as mathematical properties of functions, sets, spaces, when the problem is being solved. The more important are universal representations of AI learning models (methods) providing a combination of these models to realize hybrid approach. The aim of this paper is to describe some universal hybrid approaches in AI and their applications based mainly on the logical functions and psedo-Boolean optimization theories.

2 The setting of a problem

The main aspect of AI applications is a learning based decision making. Intelligent learning methods are used to synthesize concepts, classes, patterns as special cybernetic descriptions of real-word domains. All differences in learning (inference) methods are determined by the kind of initial information (data, knowledge) used to form concepts. The First best-kept secret of success in intelligent learning is that we must as much as possible use all relevant and reliable initial information. To a considerable extent, advantages of Hybrid methods are defined by using various kinds of initial information. But in most cases, problems arising when AI cybernetic learning methods are used, are the consequences of the initial information incompleteness.

Let X is the set of alternatives; $\{x^*\} \subset \Omega \subseteq X$, where $\{x^*\}$ is a subset of the best (optimal) alternatives and Ω is a subset of the admissible alternatives. We suppose, that all elements from $\{x^*\}$ are equivalent with respect to some goal criterion. The sets $\{x^*\}$, Ω are not given completely (accurately): there are only some initial information $I_0= I_0(\Omega,\{x^*\})$. By using this initial information I_0 we formulate the following problem.

Problem D. Find in the set X the subset $\{x^*\}$ of the best alternatives or at least one element from $\{x^*\}$ by using incomplete initial information I_0.

Let $\{I\}=\{I(\Omega,\{x^*\}\}\}$ is a set of various initial incomplete informations and I_0, I_1, ... are elements of $\{I\}$. The closure of the set $\{I\}$ under intersection and union of its elements joints and intersections is called *the possible information space* Y. We denote $\overline{N_0}$ the set of alternatives x such that presence of the information I_0 implies a truth of the predicate "$x \notin \{x^*\}$". The set $\{ N_0 = X \setminus \overline{N_0} \}$ is called *the initial region of an indeterminacy of a choice.* A solution of the problem D presupposes to synthesize an algorithm A realizing mapping A: $Y \rightarrow 2^X$. We'll denote {A} the set of all such algorithms and A_0, A_1, ... its arbitrary elements. Algorithm A is called *correct* on the information $I \in Y$, if $A(I) \subseteq \{x^*\}$ and *absolutely correct*, if $A(I) = \{x^*\}$.

If for the information I the correct (absolutely correct) algorithm A exists, then this information I is called *complete (absolutely complete)*. Otherwise the information I is called *incomplete*, and we deal with *incomplete data.* Algorithm A is called *conforming* with the information I for the problem D, if $\{x^*\} \subseteq A(I) \subseteq N_0$. If we apply the conforming algorithm A, we obtain the set $N_A = A(I_0)$ such that $\{x^*\} \subseteq N_A$. This set with the inclusion $\{x^*\} \subseteq N_A$ is some additional information I_A about the set $\{x^*\}$. The additional can supplement the initial information I_0, and we can make available the further information $\overline{I_0} = I_0 \cup I_A$.

Let $I_0 \in Y$ is some initial information. Any information $I_1 \in Y$ is called additional, if $I_0 \neq I_1 \neq \emptyset$. Additional information is called *narrowing*, if such conforming algorithm A_1 exists, that $A(I_0) \supset A_1(\overline{I_0}\, U I_1 \quad) \supseteq \{x^*\}$, where A is arbitrary conforming algorithm. We define the sequential narrowing process

$A_0(I_0) \supset A_1(\overline{I_0}\, U I_1) \supset \ldots \supset A_k(\overline{I_{k-1}}\, U I_k) \supseteq \{x^*\}$, which consists of the finding of the sequences of algorithms A_0, A_1, ..., A_k and additional informations I_1, ..., I_k, where

$\overline{I_j} = \{\overline{I_{j-1}}\, U I_j\}, j = \overline{1,k}$.

If we consider X as a features' space which is covered by collection of classes (sets) K_1, ... , K_l, then we use some initial information (cases or rules) to find class membership functions (predicates) d_j: $X \rightarrow \{0,1\}$ such that $(d_j(x)=1) \Leftrightarrow x \in K_j \subset X$: $j = \overline{1,l}$. Usually some learning procedure is used to find functions d_j. The last problem with incomplete initial information will be called **a problem Z.** Note, that problem D and Z are closely related, because the finding of the function d_j is equivalent to the finding of the set K_j which is apriori accurately unknown. So, we'll deal with two problems **D** and **Z** of an optimal decision making and a concept formation on the base of incomplete information and learning procedures.

3 Disjunctive Normal Forms as a language for unification of Hybrid System development

We denote $x_i^{\sigma i}$ a literal, $x_i^{\sigma i} = x_i$ if $\sigma_i = 1$, and $x_i^{\sigma i} = \overline{x_i}$ if $\sigma_i = 0$, where x_i is a Boolean variable or predicative symbol. The expression

$$\bigvee_{j=1}^{m} x_{j_1}^{\sigma_{j_1}} \& \ldots \& x_{j_{rj}}^{\sigma_{j_{rj}}}$$

is called a Disjunctive Normal Form (DNF), which consists of m elementary conjunctions (all variables contained in the literals of the elementary conduction are different).

In the survey paper [3] it is shown how to build the DNF which is equivalent to the binary decision tree or to the Logical Production System [4]. Any DNF D_s defines the Boolean function $D_s(x_1, \ldots, x_n)$ so that, for instance, $D_s(\tilde{X})=1$ if $\tilde{X} \in K_s$ and $D_s(\tilde{X})=0$ if $\tilde{X} \notin K_{s,}$ $s = \overline{1,l}$, i.e. D_s is the concept or the membership function of the class K_s; $\tilde{X} =(X_1,\ldots,X_n)$ This interpretation of DNFs is associated with a problem **Z.**

For the problem D the DNF D_Ω defines the characteristic function of the set Ω of admissible solutions and may be used to form an optimization model for a problem S solving. A simplest kind of the DNFs comparison is their multiplication. This approach was used for the dual expert system DUELS [5] development. In [6] we present Analogical reasoning based DNF synthesis. The Dual is called Expert System using jointly knowledge based and case based decision making by means of synthesis of the DNFs for both methods as shown above. The DNF $D_i(\tilde{X})$ can be obtained by means of empirical data generalization (empirical induction). For instance, the learning table for the DNF $D_i(\tilde{X})$ synthesis is extracted from the empirical Case Data Base included in DUELS [5]. The DNF $D_i(\tilde{X})$ is called the *Logical Description of the Inductive Region Deductibility (LDI)*. The DNF $D_d(\tilde{X})$obtained by means of the deductive inference using the set of productions contained in the Knowledge Based is called *Logical Description of the Deductive Region Deductibility (**LDD**)*.

We suppose that both $D_i(\tilde{X})$ and $D_d(\tilde{X})$ DNFs define the rules r_I and r_d for the same goal predicates $g(\tilde{X})$ such that

$$r_I\text{: } g(\tilde{X}) \text{ is true if } D_i(\tilde{X}) = 1;$$

$$r_d\text{: } g(\tilde{X}) \text{ is true if } D_d(\tilde{X}) =1;$$

Conjunction $D_d(\tilde{X})\&D_i(\tilde{X}) = D_{id}(\tilde{X})$ obtained after some simplifications is called the *Logical Description of the Combine Region Deductibility* (***LDID***). DUELS provides a computation of the ***LDID*** which defines the rule r_{Id}: $g(\tilde{X})$ is true if $D_{id}(\tilde{X})=1$;

Let $P(r_I)$ and $P(r_d)$ be the error probabilities of the rules r_I and r_d. If $P(r_I)\leq 0.1$, $P(r_d)\leq 0.1$, and these rules are statistically independent, then the error probability of the rule r_{Id} is equal to $P(r_{Id})= P(r_I)\cdot P(r_d)\leq 0.01$ and less than both $P(r_I)$ and $P(r_d)$.

4 The Canonical Model of an optimal choice

As shown in [7] any pseudo-Boolean function optimization problem with constraints which define a non-empty set of admissible solutions Ω can be represented in an equivalent form with a disjunctive constraint:

$$\operatorname{extr} F(\tilde{X}),\quad \bigvee_{j=1}^{m} x_{j_1}^{\sigma_{j_1}} \& \dots \& x_{j_{r_j}}^{\sigma_{j_{r_j}}} = 1, \qquad (1)$$

where $F:\{0,1\}^n \to \mathbf{R}$ and the constraint given in a form of Boolean equation in fulfilled when and only when $\tilde{X} \in \Omega$ - the set of admissible solutions. The model (1) is called *canonical.* The DNF in the left part of the constrain-equation can be obtained by using cases and/or knowledge depending on a kind of initial information.

To explain in detail this approach based on the canonical model let us consider the following example. Let the initial information about the goal function F consists of the knowledge:

a) F is a linear function, i. e. $F(\tilde{X})=c_1x_1+ c_2x_2+\dots+ c_nx_n+c_0$;

b) it is known: "$c_i>0$", "$c_i<0$", or "$c_i=0$" for any coefficient c_i, $i = \overline{1,n}$;

c) knowledge about constraints in the decision making problem are given in a form of a set of productions;

$$\begin{cases} x_{j1}^{\sigma_{j1}},\dots,x_{jr}^{\sigma_{jr}} \to g_j & (j=\overline{1,m}) \\ g_j \to g_0 \end{cases}$$

Let's denote $N_j = \{ \tilde{X} : x_{j1}=\sigma_{j1}, \dots , x_{jrj} = \sigma_{jrj} \}$ - an interval containing all Boolean vectors $\tilde{X}$ such that $K_j(\tilde{X}) = 1$, where K_j is the conjunction number j in DNF constraint. It is evidently that the region of the admissible solution can be expressed as $\Omega = \bigcup_{j=1}^{m} N_J$.

The extremal solution in any interval N_J for the problem max $F(\tilde{X}), \tilde{X} \in N_j \subset \Omega$, is reached in any point $\tilde{\alpha}_j^* = (\alpha^*_{j1}, \dots, \alpha^*_{jn}) \in N_j$ satisfying the following condition for any $i = 1,n$:

$$\alpha^*_{ji} = \begin{cases} \sigma_{ji}, & i \in \{j_1, \dots, j_{r_j}\} = I_j, \\ 1, & (i \notin I_j) \wedge (c_i > 0), \\ 0, & (i \notin I_j) \wedge (c_i < 0), \\ \Delta, & (i \notin I_j) \wedge (c_i = 0), \end{cases}$$

Where $\Delta \in \{0,1\}$ is arbitrary value.

In [8] we present the sufficient condition of the absolutely correct solution finding for the considering problem.

Example 1. Let the problem is given:

$$\begin{cases} \max(c_1 x_1 + c_2 x_2 + c_3 x_3); c_1 > 0; c_2 < 0; c_3 > 0; \\ x_1, \overline{x_2} \rightarrow g_0; x_1, \overline{x_3} \rightarrow g_0; \\ (g_0 = 1) \Leftrightarrow (\tilde{x} \in \Omega). \end{cases}$$

$N_1 = N_{x1\overline{x2}}$; $N_2 = N_{x1\overline{x3}}$; $N_1 \cup N_2 = \Omega$; $\tilde{\alpha}_1^* = (1,0,1)$; $\tilde{\alpha}_2^* = (1,0,0)$;

$F(\tilde{x}) = c_1x_1 + c_2x_2 + c_3x_3$; $F(\tilde{\alpha}_1^*) = c_1 + c_3 > c_1 = F(\tilde{\alpha}_2^*)$.

By using the initial information about signs of the coefficients c_1, c_2, c_3 we had found the external solution $\tilde{\alpha}_1^* = (1,0,1)$. It is evident, that if the production system defining the rule $(g_0 = 1) \Leftrightarrow (\tilde{x} \in \Omega)$ is correct, then the solution $\tilde{\alpha}_1^*$ is absolutely correct.

The Hybrid approach based on deduction and optimization joining has been shown in example 1. To improve this approach when initial information (productions) is incomplete or has inaccuracies, the cases as additional information and induction based inference can be used.

5 Applications

A. There are many papers dedicated to the portfolio selection process consisting on selection a subset of stocks for inclusion in the portfolio. The paper [9] presents an interactive procedure for learning and interpreting the choices available to a decision maker confronted with a discrete decision problem. The procedure guides a decision maker through the evaluation process and has an ability to learn from previous chooses (cases) and generates a set of rules (productions) that describe a decision maker's posterior preferences. The induction with 43 stocks finally (4 learning iterations where used) produced the set of 22 rules presented in the paper. These rules (productions) can be used to define a set Ω of admissible stocks (as shown in example 1). Any stock is described by the specific attributes or features. There are 13 attributes are selected in [9]. According to these attributes the features' predicates are built: $x_1,\ldots,x_n$. The coefficients $c_1,\ldots,c_n$ evaluating predicates' value for the linear value function $F(\tilde{x})=c_1x_1+ \ldots + c_nx_n$ can be obtained by experts. So, we have the model $\max(F(\tilde{x}))$, $\tilde{x} \in \Omega$. According this model, the best stock β_0 from the initial stocks' set P_0 can be selected. The following choices are realized from the set $P_1=P_0\backslash\{\beta_0\}$ and so on while $P_t=P_{t-1}\backslash\{\beta_{t-1}\}$ is not empty or for any $\beta \in P_t$ $\tilde{x}(\beta)\notin\Omega$.

B. The main problem of the regional ecological-economic control is a choice of the actions' package from the set of admissible actions to transfer the region's state from the critical situation. IDSS TAVRIDA is mainly intended for the solution to this problem and based on the canonical model (1). IDSS TAVRIDA contains the Data Base of ecological factors, the Data Base of ecological actions, the Data Base of ecological factors generating sources, the Data Base of experts and expert estimations, and the Knowledge Base. The critical ecological situations arising as results of ecological actions execution are described by the set of productions [10].

C. The Dual expert system DUELS [5] using jointly knowledge-based and case-based decision making by means of synthesis of the DNF's for both methods as shown above in the section 3.

References

1. Langley P., Simon H.A. Application of Machine Learning and Rule Induction. CACM, 1995, Vol. 38, N 11, 55-64

2. Rendell L. Induction as Optimization. IEEE Trans. on SYST., Man. & Cybern., 1990, Vol 20, N 2, 326-338.

3. Donskoy V. Pseudo-Boolean Scalar Optimization Models with incomplete information. GMöOR Newsletter, Germany, 1996, N 1/2, 20-25.

4. Donskoy V.I. The logical Production Systems: Analysis and Synthesis. Cybernetics and Syst. Anal., Kiev, 1994, N 4, 11-22.

5. Donskoy V.I. The Dual Expert Systems. Tech. Cybern., Moscow, 1993, N 5, 111-119.

6. Donskoy V.I. Nuriev R.E. Analogical Reasoning in Logical Hybrid Expert Systems. Proc. of 4th Init Conf. PRIP'97 "Pattern Recognition and Information Proceeding", 20-22 May, 1997, Minsk, Belarus-Poland, 1997, Vol. 1, 290-294.

7. Donskoy V.I. Pseudo-Boolean optimization with a disjunctive constraint. J. Comp. Maths. Math. Phys., Pergamon, 1994, Vol. 34, N 2, 389-398.

8. Donskoy V.I. Kozlova M.G. The Knowledge Based Optimal Decision Making Models. Proc. of 4th Int. Conf. "Knowledge-Dialogue-Solution", 15-20 Sept, 1997, Yalta, Ukraine, 1997, Vol. 2, 336-343.

9. Jog V., Michalowski W. An interactive Procedure for Learning about Preferences: Case Study of a Portfolio Manager. J. of Multi-Criteria Decision Analysis, 1994, Vol. 3, 27-40.

10. Bashta A.I., Donskoy V.I., Kuznetzova T.L. Expert systems: Perspectives of Applications for Regional Ecological Control and Planing. "Mashinna obrobka informazii', Kiev, State Univ. of Economics, 1995, Issue 56, 138-146 (In Ukrainian).

A Contextual Model of Beliefs for Communicating Agents

Pierre E. Bonzon
HEC, University of Lausanne
CH-1015 Lausanne, Switzerland
pierre.bonzon@hec.unil.ch

Abstract. We consider the problem of reasoning about the cognitive state of communicating agents. In order to compute belief revisions resulting from information exchanges, these agents could be ascribed models of explicit mental states. Towards this end, we introduce a model of distributed beliefs based on formalizing the assertion «*fact P is believed by agent X in context C*». These developments give rise to a meta-logic program embodying the various computational aspects (object class and instance creation, proof system, message interpreter) of a complete agent test bed.

1 Introduction

In order to allow for novel human-computer or computer-computer interaction, software agents could well be ascribed explicit mental states. These would then have to be regularly updated on the basis of the information exchanges taking place in a given context. We thus face the problem of reasoning about the cognitive changes resulting from communication.

Recent developments in the *theory of knowledge* have shown that communication can be viewed as the act of upgrading the state of knowledge in a distributed system [5]. At the low end of the spectrum, *distributed knowledge* of an implicit fact arises when its explicit deduction requires the gathering of information which is disseminated among the members of a group. At the high end, *common knowledge* corresponds to a fact that is publicly and with full reciprocal awareness known to all members. As each actor's performance (let it be a processor or a software agent) depends on its local state, there should be a natural driving force in the system to try and climb up the knowledge hierarchy by using available communication channels. This very first principle potentially has strong implications in many domains, including the field of intelligent environments.

This analysis has not yet had a real impact on practical applications. For a disturbing aspect of most current theories of knowledge is that they can only *ascribe* knowledge. As such, « there is no notion of the agent computing his knowledge, and no requirement that the agent be able to answer questions based on his knowledge » [5]. Explanations for this usually point at the *possible world semantics* underlying knowledge models, which in turn imply *logically omniscient agents* : agents are theoretically supposed to know all universally valid logic formulas and all logical consequences of their knowledge. Practically, however, they are usually denied effective deductive capabilities. Furthermore, common approaches to reasoning about knowledge share with classical logic a severe drawback, i.e. a lack of *locality* and/or *modularity*: in other words, reasoning within a single theory, with all the formulas equally considered for every reasoning task, is not appropriate.

After John McCarthy argued that the new logic agenda was to formalize *contexts*, make context a parameter, build context hierarchies, and then use non-monotonic reasoning for upward inheritance, new paths have been investigated. As the result of the work of Buvac and Mason, axiomatized theories for both a propositional and a quantificational logic of contexts have been proposed [3][4]. These were followed by a restricted logic of contextual implications [2], which, much in the spirit of the language of Horn clauses of logic programming, has an associated effective proof system. The next step, and goal of the present work, has been to try and build, within this new logical formalism, a model of distributed beliefs based on formalizing the assertion « *fact P is believed by agent X in context C* ».

The rest of this paper is organized as follows: in section 2 we review and extend results from [2] and come up with a model of distributed beliefs; in section 3, we propose a model of communicating agents as object with distributed beliefs; these developments give rise to a meta-logic program embodying the various computational aspects (object class and instance creation, proof system, message interpreter) of a complete agent test bed; section 4 proposes a simple example of software car drivers reasoning about each other.

2 A Contextual Model of Distributed Beliefs

Let us first review the basic concepts introduced in [2]. We shall assume a minimal background in first order logic, including the definition of a substitution θ, of an instance eθ or Eθ (of a single expression e or of a set of expressions E) and of a most general unifier (mgu) of a set of expressions. In order to define a language $\boldsymbol{L}$ of *contextual implications*, we consider a two-sorted predicate calculus (without identity), and denote by $\boldsymbol{C}$ the set of terms of the context sort, and by $\boldsymbol{A}$ the set of atomic formulas of the discourse sort. Our language $\boldsymbol{L}$ is then defined as the least set satisfying the equation $L=A\cup L\rightarrow L\cup ist(C, L)$, with modality *ist(c,p)* meaning "formula $p\in L$ is true in context $c\in C$". The subset $F\subset L$ whose formulas do not contain any implication connective defines the language of *contextual facts*

e.g.
```
on(a,b)
ist(block(S),on(X,Y) -> above(X,Y))
ist(C,P->Q) -> ist(C,P) -> ist(C,Q)
```
are expressions belonging to $\boldsymbol{L}$.

N.B We adopt the PROLOG convention of using capital letters to represent variables and following the results of ambivalent logic [7] we do not distinguish between ordinary variables (e.g. `X,Y`) standing for terms, and meta variables (e.g. `P,Q`) standing for arbitrary formulas.

In contrast to previous context formalizations, we want to be able to define *local theories* (i.e. sets of formula closed under contextual deduction, or, equivalently, deductible in a formal system of logical axioms and inference rules augmented with context-dependant non-logical axioms). This will be achieved by assuming *context assertions* whose formulas will be taken as the non-logical axioms of a local theory.

Definition (context assertion): A context assertion is a formula α:Φ where α is a term denoting a context and Φ is a a set of contextual implications, i.e. $\Phi \subset \mathcal{L}$, with all variables appearing in either α or Φ universally quantified

e.g.
```
block(s0):[on(a,b), on(b,c)].
block(S):[on(X,Y) -> above(X,Y)].
```

Given a context assertion α:Φ and a context *C*, we will say that assertion α:Φ *relates to* (or partially defines) context *C* if *C* can be unified with α

e.g. in the above example, s0 is a situation constant, and thus the first assertion relates to (or partially defines) the context of a block world in a given situation s0; in contrast, S is a situation variable, and the second assertion relates to any block world context in situation s0, s1, etc...

Definition (axiom instance): If *C* is a context and α:Φ is a context assertion related to *C*, with $C\theta = \alpha\theta$ (i.e. θ is a most general unifier of *C* and α), then any formula *P* wich can be unified with a formula $\phi\theta \in \Phi\theta$ is called a (non-logical) *axiom instance* for context *C*.

e.g given the above context assertions related to `block(s0)`,

```
on(a,b), on(b,c)
on(X,Y) -> above(X,Y)
```

are axiom instances for context `block(s0)`.

N.B. If *instance(C,P)* means "*P* is an axiom instance for context *C*", then we have the PROLOG definition `instance(C,P):- C:X,member(P,X).`

Let us now consider the following axiom system for contextual deduction including a link to the non-logical axioms of a theory contained within explicit context assertions:

logical axioms	
ist(C, P→Q) → ist(C, P) → ist(C,Q)	***distribution axiom (or axiom K)***
ist(C,P) → ist(C_1,ist(C,P))	***semi-flatness axiom***
inference rules	
from instance(C,P) *infer* ist(C,P)	***link to non-logical axioms***
from P *and* P → Q *infer* Q	***modus ponens***

Fig. 2.1 Axiom system for contextual deduction

Ignoring the link to non-logical axioms, this system is obtained from normal multimodal logic K_n after substituting the semi-flatness axiom for the usual necessitation rule «*from p infer* $K_i p$ »: whereas in system K_n formula $K_i p$ means « *agent i knows p* » and thus requires *p* to be valid (i.e. true in all models) for the agent to know it, formula *ist(C,P)* is interpreted as truth in context and therefore *ist(C_1,ist(C,P))* is true whenever *ist(C,P)* happens to be true. Lacking a model semantics for this logic, we cannot state any soundness and/or completeness property. Let us mention however that this system has been proved [2] equivalent to a system of *natural deduction* in context based on the intuitive notion of *entering* and *leaving* a context, as originally suggested by McCarthy [8]. Equivalently, this axiomatic system can be given by the following Prolog program :

```
ist(C,P->Q) -> ist(C,P) -> ist(C,Q).        distribution axiom
ist(C,P) -> ist(C1,ist(C,P)).               semi-flatness axiom

ist(C,P):- instance(C,P).                   link to non-logical axioms
Q:- P->Q,P.                                 modus ponens
```

Fig. 2.2 Logic program for contextual deduction

This Prolog program, which can be run on any standard Prolog system, actually represents a meta-logic program constituting an interpreter for the language of contextual deduction : whereas the axioms in the programs are unit clauses (i.e. Prolog structures, or facts, with "->" ,":" and "." declared operators), the inference rules are definite clauses.

N.B. Running program 2.2 in a system which, such as Prolog, relies on a uniform depth first search strategy will systematically get into infinite recursions. As a way out, one may implement an *iterative deepening search* of 2.2 as follows:

```
search(P/N):- P/N;
              N1 is N+1,
              search(P/N1).
```

with

```
(ist(C,P->Q) -> ist(C,P) -> ist(C,Q))/N.
(ist(C,P) -> ist(C1,ist(C,P)))/N.
ist(C,P)/N :- instance(C,P).
Q/N :- N>0, N1 is N-1,
       (P->Q)/N1, P/N1.
```

and the new top level call

```
ist(C,P) :- search(ist(C,P)/1).
```

We conclude this review with the definition of a *contextual deduction relation*.

Definition (contextual deduction relation $\vdash$**):** A fact P is said to be derivable in context C, noted $C \vdash P$, iff $\exists$ a derivation (in the usual sense) of *ist(C,P).*

Going a step further, we now consider a logical model of an important concept for reasoning about the cognitive state of agents, i.e. that of *belief.* Beliefs, as opposed to knowledge, are not necessarily justified, i.e. may actually be false. Within the spectrum of knowledge states alluded to in the introduction, we shall distinguish two fundamental cases, i.e.

a) *private* beliefs, which refers to facts that are believed by some or all members in a group without reciprocal awareness

b) *public* beliefs, corresponding to common knowledge and thus refering to facts that are believed by all members in a group with full reciprocal awareness (i.e. « everybody believes », « everybody believes that everybody believes », etc.).

Private and public beliefs can be either *implicit* or *explicit.* In our formalism, explicit beliefs sets are given under the form of context assertions. Implicit beliefs follow from deductions involving explicit beliefs. Finally, *distributed* beliefs are implicit beliefs whose deduction requires external (as opposed to one's own) private beliefs.

External beliefs follow from public beliefs expansion, or imply an act of *communication* (e.g. by having agents, implemented as *object* instances, send messages to other agents, as developped in our next section).

As a way to express private beliefs, let us introduce a new modality *bel(X,P)* meaning « agent *X* believes *P*». In analogy with the definitions given in the first paragraph of this section for the *ist(C,P)* modality, we define the notion of *nested belief implications* and *nested beliefs* by substituting *bel* for *ist* in the definitions of *contextual implications* and *contextual facts*, respectively. Let us next consider the following context assertion containing axioms for this new modality:

```
belief:
[public(P) -> bel(X,P).                      axioms for public beliefs
 public(P) -> public(bel(X,P)).              expansion

 bel(X,P) -> bel(X,bel(X,P)).                axiom for positive introspection
 bel(X,P->Q) -> bel(X,P) -> bel(X,Q).        distribution axiom

 (bel(X,R) -> bel(X,S) -> bel(X,T))          axiom for nested distribution
    -> bel(Y, bel(X,R))
        -> bel(Y, bel(X,S))
            -> bel(Y,bel(X,T))].
```

Fig. 2.3 Axioms for beliefs

Public beliefs are expressed via an additional modality *public(P)* meaning « *P is a public belief* ». The first two axioms allow for the expansion of public beliefs into nested private beliefs of arbitrary nesting depth (i.e. formulas such as *bel(X, bel(Y,P))*, *bel(X,bel(Y,bel(Z,P)))*, and so on, will follow from formula *public(P)*). The axiom for positive introspection reflects an intuitive property of beliefs. The axiom for nested distribution allows for the derivation of nested distribution axioms, leading themselves to nested beliefs of arbitrary nesting depth.

Clearly the above context assertion defines a *belief* theory. If explicit public and/or private beliefs were included in this context assertion, program 2.2 could be used to derive *ist(belief,bel(X,P))* formulas, and from there new explicit beliefs *bel(X,P)*

e.g. if
```
bel(john, loves(mary,john) -> happy(john))
public(loves(mary,john))
```
were to be included in the *belief* context, then
```
bel(john,happy(john))
```
could be derived as new explicit belief.

Additional links are required however if explicit private and/or public beliefs are available in *separate* context assertions. As we shall see below, it will prove convenient to associate two contexts (i.e. *C(X)* and *C(public)* for the private and public beliefs of agent X, respectively) to any given multi-agent system considered globally as a single *belief context C*

e.g.
```
c(john):[loves(mary,john) -> happy(john)].
c(public):[loves(mary,john)].
```
could be instead given as explicit context assertions.

Towards this goal, let us replace the unique link `ist(C,P):- instance(C,P)` of program of fig. 2.2 by the following set of links:

```
ist(C,P):- instance(belief,P).                   link to belief theory
ist(C,bel(X,P)):-instance(C(X),P).               link to private beliefs of X in context C
ist(C,public(P)):-instance(C(public),P).         link to public beliefs in context C
```

Fig. 2.4 Links for contextual belief deduction

As a result we get the following definition of a *distributed belief deduction relation*:

Definition (distributed belief deduction relation $\vdash_B$): A nested belief P is said to be believed by agent X in belief context C, noted $C \vdash_B bel(X,P)$, iff $\exists$ a derivation of *ist(C,bel(X,P))* using the links of fig. 2.4.

3 A model of agents as objects with distributed beliefs

As discussed above, the deduction of distributed beliefs possibly implies pooling together, through *communication*, disseminated private beliefs. Toward this end, we require a model of communicating agents, which we informally introduce as follows:

- agent are object instances belonging to one of many disjoint object classes
- agents of a given class share the explicit public beliefs attached to their class; in addition, agents can have their own private beliefs
- agent's reasoning capabilities allow them to compute implicit beliefs
- agents communicate by sending messages to other agents in the same class
- messages can request agents to privately or publically express their beliefs, which then will become either private or public new explicit beliefs

Formally, we propose the following definitions:

Definition (agent class and instances): a class *Class* of communicating agents is an object encapsulating two contexts, i.e.

- a context *Class(public)* defining the class public beliefs
- a context *Class(methods)* defining the requests conveyed by messages ;

in addition, each agent instance *X* of class *Class* gives rise to an object comprising

- a context *Class(X)* defining the agent's own private beliefs.

N.B. As contexts are made explicit through context assertions, whenever we mention a context *C*, we actually refer to any context assertion *C:_*.

Definition (class methods and messages): methods, asserted within context *Class(methods)*, have the format *method(Class.Call,Body)*; messages sent to agents have the format *Class.Call* ; they are interpreted by the procedure

```
Class.Call:-instance(Class(methods), method(Class.Call,Body)),
            execute(Body).
```

with

```
execute([]).
execute([H|T]):- call(H), execute(T).
```

Each call contained within the body of a message can be any Prolog call or messages of format *Class.Call*, including *Class.bel(X,P)* for belief deductions *within classes* (see below). Following the traditional object-oriented paradigm, both agent classes and instances are created by sending messages to appropriate objects. Class creation follows from sending the message *metaclass.newClass(Class)* to a predefined object encapsulating a context *metaclass(methods)* defined as follows:

```
object(metaclass(methods):
    [method(metaclass.newClass(Class),
            [class(Class(methods):Methods),
             assert(object(Class(methods):Methods)),
             assert(object(Class(public):[]))])]).
```

Fig. 3.1 Metaclass object

As it can be seen in the body of *newClass*, this creation involves first the instantiation of a context assertion *Class(methods):Methods* from a class template (see below fig. 3.2), followed by its encapsulation, together with an empty context assertion *Class(public):[],* into a new asserted object.

N.B. Following the possible encapsulation of belief contexts into objects, the definition of *instance(C,P)* now includes a disjunction

```
instance(C,P) :- (C:Y ; object(C:Y)),member(P,Y).
```

In contrast, the creation of an instance is taken care of by its class. Towards this end, the class template (actually encapsulating a context *Class(methods),* see fig 3.2) contains a method *Class.newAgent(X),* which simply encapsulates an empty context assertion *Class(X):[]* within a new asserted object. Other methods contained in the class template given below include the methods *Class.add(X,Belief)* and *Class.assume(X,Context)* for updating belief sets. Method *Class.bel(X,P)* involves the iterative deepening search of the distributed belief deduction relation.

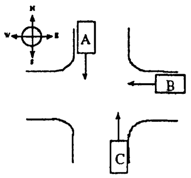
Finally, *Class.askAndTell(Agent,bel(X,P))* is an example of a method for the agents to communicate: in this case, agent *X* of class *Class* is asked to report his beliefs about fact *P* to agent *Agent* (of the same class); this is achieved by first collecting all such beliefs into a set (using Prolog's built-in *setof* predicate); depending on this result, it will then add into *Agent*'s belief set either each collected belief *bel(X,P)* (using Prolog's built-in *forall* predicate) or a single negation *not bel(X,P).* This method is thus an example of explicit negation introduction relying on negation by default.

```
class(Class(methods):
    [method(Class.newAgent(X),
            [assert(object(Class(X):[]))]),
     method(Class.add(X,Belief),
            [object(Class(X):Beliefs),
             append([Belief],Beliefs,NewBeliefs),
             retractall(object(Class(X):_)),
             assert(object(Class(X):NewBeliefs))]),
     method(Class.assume(X,World),
            [World:Assumptions,
             forall(member(Belief,Assumptions),
             Class.add(X,Belief))]),
     method(Class.bel(X,P), [search(ist(Class,bel(X,P))/1)]),
     method(Class.askAndTell(Agent,bel(X,P)),
            [(setof(P,Class.bel(X,P),Set) ->
              forall(member(P,Set),
                     (Class.add(Agent,bel(X,P)),
                      write(X),write(' believes '),write(P)));
             Class.add(Agent,not bel(X,P)),
             write(X),write(' does not know'))])]).
```

Fig. 3.2 Class template

4 Example

Let us consider the following example of a intelligent environment involving software car drivers reasoning about each other (as originally presented in [1]).



There are no signs or traffic lights at this crossing, so drivers must give way to cars coming on their right side. Thus A can pass, while B and C must wait. However, C can further reason that B will wait. Hence C will conclude that he can pass.

A straightforward modeling of this problem leads to the following definitions:

```
cars:[car(a,north),car(b,east),car(c,south),nocar(west)].

priorities:[car(X,north)-> car(Y,east)  -> priority(X,over(Y)),
            car(X,west)  -> car(Y,north)-> priority(X,over(Y)),
            car(X,south)-> car(Y,west)  -> priority(X,over(Y)),
            car(X,east)  -> car(Y,south)-> priority(X,over(Y)),
            car(X,east)  -> nocar(north)-> nopriority(over(X)),
            car(X,north)-> nocar(west)  -> nopriority(over(X)),
            car(X,west)  -> nocar(south)-> nopriority(over(X)),
            car(X,south)-> nocar(east)  -> nopriority(over(X))].

actions:[nopriority(over(X)) -> do(X,pass),
         priority(Y,over(X)) -> do(Y,pass)-> do(X,wait),
         priority(Y,over(X)) -> do(Y,wait)-> do(X,pass)].
```

N.B. In the absence of a negation operator, negative information must be dealt with explicitly through the use of additional predicates (e.g. `nocar`, `nopriority`). Implications associate to the right, so that $A \rightarrow B \rightarrow C$ reads $A \rightarrow (B \rightarrow C)$, i.e. $(A \wedge B) \rightarrow C$.

Each of these three context assertions corresponds to public belief sets. As a result, we get the following scenario for setting up agents:

```
metaclass.newClass(drivers).
drivers.newAgent(a).
drivers.newAgent(b).
drivers.newAgent(c).
drivers.assume(public,cars).
drivers.assume(public,priorities).
drivers.assume(public,actions).
```

Implicit driver's beliefs (or intentions) follow from the following messages:

```
drivers.bel(a,do(a,Action)).
 Action = pass.

drivers.bel(b,do(a,Action)).
 Action = pass.

drivers.bel(b,do(b,Action)).
 Action = wait.
```

```
drivers.bel(c,do(b,Action)).
 Action = wait.
drivers.bel(c,do(c,Action)).
 Action = pass.
```

... and so on.

Assuming yet another car leads to a deadlock (i.e. no implicit belief is found)

```
drivers.newAgent(d).
drivers.delete(public,nocar(west)).
drivers.add(public,car(d,west)).

drivers.bel(a,do(a,Action)).
 no
drivers.bel(b,do(a,Action)).
 no
drivers.bel(b,do(b,Action)).
 no
```

... and so on.

Adding a new public rule and asking somebody, say d, to publicly express his intentions will unlock the situation in favor of c. We first give

```
drivers.add(public,priority(Y,over(X))
                    -> not bel(Y,do(Y,pass))
                           -> do(X,pass)).
```

This new rule indicates that even though Y has priority over X, if Y has expressed no intention to pass then X can intend to pass. We then request

```
drivers.askAndTell(public,bel(d,do(d,Action))).
 d does not know
```

After d publicly expressed (by default) no intention to pass we can then request c to express his own intention to pass

```
drivers.askAndTell(public,bel(c,do(c,Action))).
 c believes do(c,pass)
```

5 Conclusions

We proposed a model of communicating agents relying on a distributed belief deduction relation. It is worth mentionning the analogy between this deduction relation (which we gave without any support from a model semantics), and Jiang's [6] proposal for a "Multiagent Autoepistemic Logic". In his so-called "extrospective view", Jiang allows for Moore's autoepistemic *L* operator to stay outside the scope of indexed epistemic formulas, as in *L Bel(John,P)*. Clearly, our own formulas *ist(C, bel(john,P))* look like Jiang's formulas further indexed by a context C. Whereas Jiang's developments do not consider public beliefs, it thus appears that contextual deduction is required in order to account for public belief expansions *within a given multi-agent system* considered as a belief context C.

The various computational aspects of our model are embodied in a meta-logic program defining the operational semantics of a simple agent-oriented language. This model can serve as a formal clarification tool for various embedding and/or equivalence claims about *agent-oriented programming* (AOP) which were made earlier, in particular by Shoham [10], i.e.

- AOP lies in the realm of Multiple Agents Systems
- AOP can be viewed as a form of Logic Programming
- AOP can be viewed as a form of Object Oriented programming
- AOP can be viewed as a form of Knowledge-based Programming.

Numerous extensions could follow the same pattern, e.g. by encapsulating additional contexts within agent instances and/or classes. As an example, a context *Class(private)* containing assertions about the state of the world could be associated with any class. Agents could then be programmed to selectively inherit assertions reflecting their own partial view of the world (as in the classical « three wise men » puzzle, where each agent can see each other's face, but not his own).

Further work should aim at the introduction, within this formalism, of additional constructs, such as desires and intentions, and their integration with agent activities, such as planning, execution and monitoring, as realized for instance by Rao [9].

References

1. J. Barklund, K. Boberg, P.Dell'Acqua and M.Veanes, Meta-Programming with Theory Systems, in: *Meta-Logics and Logic Programming,* K. Apt and F. Turini (eds.), MIT Press 1995
2. P. Bonzon, A Reflective Proof System for Reasoning in Contexts, Proc. 14th Natl. Conf. on Artificial Intelligence (AAAI97), 1997
3. S.Buvac, V.Buvac and I.A. Mason, Metamathematics of contexts, Fundamenta Informaticae 23 (3),1995
4. S. Buvac, Quantificational Logic of Context, Proc. 13th Natl. Conf. on Artificial Intelligence (AAAI96), 1996
5. R. Fagin, J. Halpern, Y Moses and M. Vardi, *Reasoning About Knowledge*, MIT Press, 1995
6. Y.J. Jiang, On Multiagent Autoepsitemic Logic – an extrospetive view, Proc. 4th Intl. Conf. On Principles of Knowledge Representation and Reasoning (KR'94)
7. M. Kalsbeek and Y. Jiang, A Vademecum of Ambivalent Logic, in: *Meta-Logics and Logic Programming,* K. Apt and F. Turini (eds.), MIT Press 1995
8. J. McCarthy, Notes on Formalizing Context, Proc. 13th Intl. Join Conf. on Artificial Intelligence (IJCAI93), 1993
9. A.S. Rao, AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language, in: *Agents Breaking Away* (MAAMAW '96), W. Van der Velde and J.W. Perram (eds), LNAI (vol 1038), Springer Verlag, 1996
10. Y. Shoham, Agent-Oriented Programming, Artificial Intelligence, 60, 1993

Context-Mediated Behavior for AI Applications*

Roy M. Turner

Department of Computer Science, University of Maine, Orono ME 04469, USA
rmt@umcs.maine.edu

Abstract. Artificial intelligence (AI) applications are often faulted for their brittleness and slowness. In this paper, we argue that both of these problems can be ameliorated if the AI program is *context-sensitive*, making use of knowledge about the context it is in to guide its perception, understanding, and action. We describe an approach to this problem, *context-mediated behavior* (CMB). CMB uses *contextual schemas* (c-schemas) to explicitly represent contexts. Features of the context are used to find the appropriate c-schemas, whose knowledge then guides all aspects of behavior.

1 Introduction

Artificial intelligence (AI) applications are often faulted for being slow and brittle. Part of their slowness stems from the inherent difficulty of their tasks, but part also is due to their need to determine which knowledge is applicable. Brittleness, or non-graceful degradation of performance, arises in part from applying knowledge that is inappropriate for the context. These problems are exacerbated if the program is required to operate in many different contexts, performing a variety of tasks.

Humans are able to cope with real-time constraints while carrying out complex tasks in a wide range of contexts. We do this in part by being *sensitive* to the context. It is well-accepted, for example, that context plays an important role in human perception, decision-making, and social interaction.[1] Upon recognizing his or her context, a person immediately has available a great deal of information about what to expect, how to interpret what is sensed, and how to behave. For example, upon entering a theater, a person knows what to expect to find there (seats, refreshment counters, ticket offices, etc.), what actions are appropriate to achieve goals (e.g., buy refreshments rather than serving oneself), and general characteristics of appropriate behavior for the context—one behaves qualitatively different in a theater than, say, at a soccer match. This contextual

* This work was funded in part by the United States National Science Foundation under grant BES-9696044.

[1] See [5] for a discussion.

knowledge is available immediately, automatically, and effortlessly. As the context changes, new contextual knowledge is always at hand. The result is that a person's behavior is usually appropriate for the context, and behavioral decisions can be made quickly based on readily-available contextual knowledge.

In this paper, we discuss what is needed to endow an AI application with context-sensitive reasoning abilities. We then describe our approach to this problem, called *context-mediated behavior* (CMB), which uses *contextual schemas* [6, 7] to represent an agent's contextual knowledge. Contextual schemas (c-schemas) are stored in a conceptual memory from which they can be retrieved based on features of the current context. They are merged to form a coherent picture of that context, then knowledge from the resulting *context structure* is used to control all aspects of the AI program's behavior. CMB is being implemented in ECHO (Embedded Context-Handling Object), the context manager for the Orca program [7, 8], a mission-level controller for autonomous underwater vehicles (AUVs).

2 Requirements for Context-Sensitive Behavior

Explicit representation of contextual knowledge. An AI agent[2] needs to be able to represent knowledge about contexts in which it may find itself. All AI systems encode some contextual knowledge: antecedent clauses in rule-based systems, preconditions in planners, etc. However, with a few notable exceptions (e.g., [1, 3, 6]), contexts are not treated as objects in their own right. This leads to redundant representation of contextual knowledge, inability to make inferences about the context itself, and unnecessary effort to find appropriate contextual knowledge to apply to the context.

By representing contexts explicitly, the agent can use features of contexts it knows about as the basis to identify the context it is in. It can make predictions about unseen features of the current context based on its *a priori* contextual knowledge. This information can also help it disambiguate or otherwise interpret sensory data and fine-tune the meaning of its conceptual knowledge to fit the context.

An agent should also associate *prescriptive* knowledge with its context representations so that it knows how to behave in the corresponding contexts. When the context is recognized, the agent will then instantly and effortlessly have available the information necessary to behave appropriately in that context. For example, the contextual knowledge about being "in a harbor" should keep the agent from going too near the surface (there is likely to be surface traffic) or too deep (the water is likely to be shallow). Should there be an emergency, the contextual knowledge should immediately suggest landing on the bottom rather than surfacing (since that would risk a collision). Prescriptive knowledge an agent needs about a context includes: event-handling information; goal-achievement information; attention-focusing information; and parameters and goals that should

[2] In the remainder of the paper, "agent" will be used to refer to any type of AI application program.

become effective automatically upon entering or leaving a context ("standing orders").

Contextual knowledge should be clustered. When a context is recognized, all relevant knowledge should be automatically "brought to mind". This avoids inferences or other work needed to collect the knowledge at run-time. Moreover, the knowledge remains available until the context changes, thus allowing rapid, context-appropriate decision-making. As discussed below, CMB uses frame-like knowledge structures called contextual schemas, or c-schemas, to do this.

A problem is choosing which contexts to represent. We define a context to be any identifiable configuration of features (environmental, mission-related, or agent-related) that has predictive power for the agent's behavior [5]. However, it would be unwise to try to represent all such contexts. The number would be immense, and the agent would be unlikely ever to encounter the vast majority of them. A guideline to reduce this number is: represent a context as a c-schema only if: (1) it cannot be represented by merging the knowledge contained in existing c-schemas or (2) such a merger fails to prescribe the correct behavior for the context [5].

Still, the number of possible contextual schemas is too large. One possible solution is to abstract the features of the context represented, e.g., by using variables or constraints rather than actual values. This allows a single c-schema to represent many different contexts. Another possibility is to abstract entire contexts. For example, the context "in Bar Harbor during outgoing tide with low power" could be considered an instance of more abstract contexts, such as "in Bar Harbor" or "in a harbor", "outgoing tide", and "low power". Such abstractions can be combined in different ways to represent many different contexts.

This also allows the agent to handle novel contexts. A new context may be adequately represented by a single abstract context, or the agent may need to piece together several existing contexts. For example, if an AUV has never had a sonar failure in a harbor before, it might still merge the context representations for "in a harbor" and "sonar failure" to decide how to behave.

Context recognition. To make use of contextual knowledge, an agent must have a mechanism to recognize the current context. This task is essentially diagnosis: given a set of known contexts (c-schemas), use the features of the current context to determine which it is an instance of. Thus, diagnostic knowledge must be associated with context representations.

Mechanism for tracking context change. An agent must be able to determine when its current representation of the context no longer matches the actual context. Context representations can help by describing situations in which they no longer fit the current context. For example, the context "in a harbor" might state that if the water is very deep, then an AUV is no longer in the context. It is unlikely that all such context transitions can be handled this way, however. An agent will need periodically to check its context representation against the world.

Mechanism for changing contextual knowledge. As the agent carries out its tasks, it will learn about new contexts and new things about old contexts.

Consequently, an agent needs a way to update its contextual knowledge, e.g., by learning.

3 Context-Mediated Behavior

Context-mediated behavior is part of the AI technique called ***schema-based reasoning*** [7], in which packets of knowledge representing patterns in problem solving are retrieved from a conceptual memory and interpreted by a reasoner to guide its behavior. For CMB, these packets are contextual schemas representing contexts. CMB is being implemented in ECHO, the Embedded Context Handling Object. ECHO is a module in Orca, a schema-based mission controller for autonomous underwater vehicles [7, 8]. An overview of ECHO's context-management process is shown in Fig. 1.

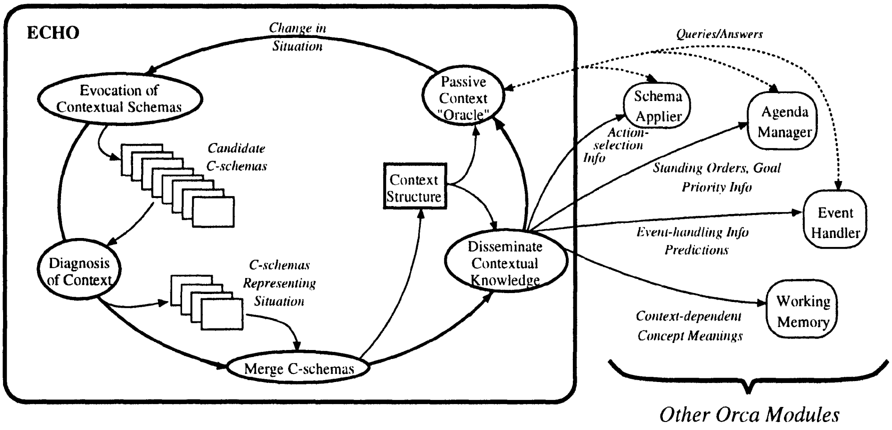


Fig. 1. ECHO.

3.1 Context Recognition

CMB's first task is recognizing the agent's current context. This is a diagnostic task. Features of the agent's current context, as reflected in sensory and other information, are compared to features predicted by known contexts, which are represented as contextual schemas. For example, `c-harbor`, a c-schema representing being in a harbor, would predict that the water column is shallow, there is surface traffic present, and so forth (see Fig. 2).

The overall diagnostic process is based on an abductive reasoning mechanism [4]. ECHO uses a conceptual memory to *evoke* candidate c-schemas based on the features of the context. In ECHO's memory, each c-schema is a memory structure that indexes related c-schemas (generally specializations of itself) based

Context description:

Actors: an AUV, other vessels

Objects: shore, bottom, surface, bottom clutter, buoys,...

Setting: shallow water, surface traffic, ...

Standing orders:

restrict depth envelope
restrict velocity envelope
turn on obstacle avoidance sonar

Events:

obstacle=>avoid to side; power failure=>abort;
in channel & detect ship=>move out of channel;...

Goals:

abort=>land and release buoy; GPS fix (do not attempt);
send message=>send message using acoustic modem;...

Fig. 2. Some of the information in c-harbor, a c-schema representing "in a harbor".

on features distinguishing them from itself [7]. Features that are different become indices that allow discrimination between c-schemas. Features are abstracted as much as possible so that similar differences point to the same specialized c-schema. When entering a new context, its features are presented to the memory, which traverses the indexing structure and returns a set of c-schemas similar in some way to the context.

Once evoked, the candidate c-schemas are combined into *logical competitor sets* (LCSs; see [2]) based on the observed features they explain. Each LCS is then solved by differential diagnosis using its c-schemas' predictions and how strongly they are evoked by the context. The c-schemas solving each LCS are merged to create the context structure.

3.2 Perception and Understanding

The context structure contains knowledge to aid perception and understanding. Predictions about unseen features of the context can bias interpretation of sensory information toward those features and can help prepare the agent for the occurrence of an unanticipated event.

In addition, the context structure contains information about context-dependent meaning of concepts [9]. For example, Orca's c-schemas contain context-dependent meanings of fuzzy linguistic variables. Orca uses a simple version of fuzzy logic in which *linguistic variables*, such as "depth", take on one of a set of *linguistic values*, such as "shallow" (e.g., [10]). The meaning of each linguistic value is specified by a membership function that maps an actual ("crisp") numeric value to a value in the range [0,1], indicating its degree of membership in the fuzzy set referred to by the linguistic value.

By storing the mapping in c-schemas, the context-dependent meaning of linguistic values can be represented. For example, a c-schema representing "in a harbor" would suggest a different meaning for the "nominal" depth than would a c-schema representing "in the open ocean". In the former, the default nominal depth should be the middle of the water column (to avoid surface traffic and bottom clutter), while in the latter, remaining near the surface would likely be best. When c-schemas are merged to form the context structure, the fuzzy values' meanings are merged as well (see [9]).

3.3 Handling Events

One of the most important things that an AI application must do is handle unanticipated events rapidly and appropriately for the current context. Some events may be predicted quite accurately. For example, the catastrophic failure of an AUV can be predicted if it is in the context of "leaking". Other events are in a sense unanticipated: e.g., a catastrophic failure *could* occur in the context of "in a harbor", but it is not expected. However, it is critical to know the appropriate (context-specific) response should the event occur.

In CMB, c-schemas contain knowledge about both kinds of events. Knowledge about a predicted event can be used to plan for the event's occurrence as well as to detect when a predicted event has *not* occurred. Knowledge about unanticipated events allows them to be quickly recognized and handled appropriately for the context. A c-schema contains knowledge about an event only when the way that event is handled is affected by the context.

Three kinds of information are recorded for each event: how to detect it, how to estimate its importance, and how to respond to it. In Orca, event detection information is represented as fuzzy rules that are given to its Event Handler. Event importance information is represented as importance estimates as well as fuzzy rules that tailor the importance to the particulars of the context. Event response information suggests a goal to activate when the event occurs. The event is handled when Orca selects the goal as its focus of attention. This cleanly integrates event handling with normal goal achievement.

3.4 Achieving Goals

Contextual schemas contain information about how important particular goals are in the context and about context-dependent ways to achieve them. The former is used to help focus attention. For example, in most contexts, an AUV's periodic goal of identifying its location (e.g., via surfacing for a global positioning system fix) is fairly important. However, in the context of rescuing a diver who is using a sonar transponder to signal for help, the goal of determining location would be much less important. In Orca, importance information is represented by priority estimates and fuzzy rules for fine-tuning the estimate based on the particular context. These are used by its Agenda Manager to focus attention. Information about how to achieve goals is in the form of suggestions about which procedural schemas (p-schemas) to use [7].

3.5 Standing Orders

Each c-schema contains a set of things that should be done when entering the corresponding context, that should be in effect when in that context, or that should be done when exiting the context. These "standing orders"[3] are either goals to activate/deactivate or parameters to set. CMB modifies behavior automatically by activating or deactivating standing orders when entering or leaving a context. For example, a c-schema for "in a harbor" would inform an AUV not to go too near the surface nor too deep. Similarly, a c-schema for docking would turn off obstacle avoidance behavior.

This mechanism allows an agent to modulate most aspects of its behavior based on its context, not just those associated with particular activities such as goal achievement or handling unanticipated events. Such automatic "background" behavior modulation can be very important in ensuring behavior that tracks the agent's evolving context appropriately.

4 Conclusion and Future Work

Context-sensitive behavior is necessary if an AI program is to rapidly modify its behavior to fit its context. Context-mediated behavior is an approach to this problem that explicitly represents contextual knowledge. Its contextual schemas are retrieved based on the context's features, then information from them is merged to guide all facets of the agent's behavior while in the context. This knowledge remains available until the context changes, thus allowing rapid, context-appropriate decision-making.

At the current time, implementation is in progress for the version of CMB described above. This implementation is embodied in Orca's context-management object, ECHO. We anticipate a complete, initial version of ECHO at or near the time of publication. Future versions will refine ECHO's knowledge representation and reasoning methods based on experience using the initial version. Initially, CMB will be evaluated via simulation experiments using Orca; ultimately, we plan to conduct in-water experiments.

Beyond that, future work will focus primarily on those requirements discussed above that are not currently part of ECHO's design. Contextual schema merger and detecting context change, for example, are difficult problems that will not be completely solved in the initial version of ECHO. Learning contextual knowledge is an important long-term goal of the work, since this will let an agent tailor its contextual knowledge over time to the particular set of contexts in which it operates.

We expect CMB to be useful for a wide range of AI applications. This includes not only other planning systems, but also other kinds of AI systems, such as rule-based systems and neural networks. For a rule-based system, c-schemas would contain rules. This has been suggested before [1], but CMB would add sophisticated c-schema retrieval mechanisms, c-schema merger, and the standing orders mechanism to modulate the application's behavior. A c-schema in a neural network-based system would contain weights appropriate for the network when operating in the corresponding context. When the application encounters an

[3] Term due to D.R. Blidberg.

instance of the context, the weights, and hence, the network's behavior, would change automatically to fit the context. We plan to explore these kinds of uses of CMB in future work.

References

1. Aikins, J.S.: A representation scheme using both frames and rules. In: Buchanan, B.G., and Shortliffe, E.H. (eds.): *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project.* Addison–Wesley Publishing Company, Reading, Massachusetts (1984) 424–440
2. Feltovich, P.J., Johnson, P.E., Moller, J.A., and Swanson D.B.: LCS: The role and development of medical knowledge and diagnostic expertise. In: Clancey, W.J., and Shortliffe, E.H. (eds): *Readings in Medical Artificial Intelligence.* Addison–Wesley Publishing Company, Reading, Massachusetts (1984) 275–319
3. Guha, R., and Lenat, D.B.: Cyc: A midterm report. *AI Magazine* **11** (3) (1990) 32–59
4. Miller, R.A., Pople, H.E., Jr., and Myers, J.D.: INTERNIST–1, an experimental computer-based diagnostic consultant for general internal medicine. *New England Journal of Medicine* **307** (1982) 468–476
5. Turner, R.M.: Context-mediated behavior for intelligent agents. *International Journal of Human–Computer Studies* (in press)
6. Turner, R.M.: When reactive planning is not enough: Using contextual schemas to react appropriately to environmental change. In: *Proceedings of the Eleventh Annual Conference of the Cognitive Science Society*, Detroit, MI (1989) 940–947
7. Turner, R.M.: *Adaptive Reasoning for Real-World Problems: A Schema-Based Approach.* Lawrence Erlbaum Associates, Hillsdale, NJ (1994)
8. Turner, R.M.: Intelligent control of autonomous underwater vehicles: The Orca project. In: *Proceedings of the 1995 IEEE International Conference on Systems, Man, and Cybernetics.* Vancouver, Canada (1995)
9. Turner, R.M.: Determining the context-dependent meaning of fuzzy subsets. In: *Proceedings of the 1997 International and Interdisciplinary Conference on Modeling and Using Context (CONTEXT-97)*, Rio de Janeiro (1997)
10. Zadeh, L.A.: Fuzzy logic, neural networks, and soft computing. *Communications of the ACM* **37** (3) (1994) 77–84

A Context-Sensitive, Iterative Approach to Diagnostic Problem Solving

Pinar Öztürk

SINTEF
Telecom and Informatics
N-7034 Trondheim, Norway
pinar@informatics.sintef.no

Department of Computer and Information Science
Norwegian University of Science and Technology
N-7034 Trondheim, Norway
pinar@idi.ntnu.no

Abstract. In domains where the knowledge is not represented by strong theories or where the problems are to be solved with incomplete information, problem solving needs to be a context sensitive process. This paper presents a task-centered methodology that we used when modelling the context in a diagnostic process. We identify two aspects of the notion of context as important: its role and its elements. We argue that a systematic investigation of the notion of context needs to be organised along these two aspects. Regarding the role that the context play we distinguish between two basic issues. These are related to its role in shifting the focus of attention , and in capturing the focus of attention. We discuss a diagnostic problem solving model, referred to as context-sensitive iterative diagnosis (ConSID) approach which is particularly appropriate for open and weak domains. We presented an implementation of this approach as a hybrid case-based and explanation-based reasoning system.

1 Introduction

The notion of context crops up every now and then in everyday life as well as in the research world. For example, as has often been referred to, the meaning of an ambiguous word depends on the context in which it is used. Another example is that our behaviour is context dependent; the way of getting food depends on where we are, and is rather different in fast food restaurants than in expensive restaurants.

The notion of context has been investigated in various disciplines such as linguistics, psychology and artificial intelligence (Brezillon, 1993; Abu-Hakima and Brezillon, 1995). Researchers in linguistics and communication have studied the effects of context on the interpretation of an utterance. Their aim was understanding how the meaning of an utterance changes when the context of the utterance changes. Researchers in psychology have studied the context effects on various cognitive processes such as perception, reasoning and learning. AI researchers have recently been concerned with modelling and representing context in connection with developing computational artifacts. However, these various researches attach different meanings to the term context. This may be a sign that different disciplines study a particular aspect of this notion. In this research we tried to take benefit of the studies on context in different disciplines when we attempt to give a particular meaning to the term context in the AI context.

Our aim, in general, is to understand the role of context in problem solving. For this purpose, we adopted the strategy of focusing on complex tasks which are frequently encountered in the real world either in everyday life or in scientific research. Context comes to the scene typically in situations where the task domain is weak, such as medical problem solving. In such domains problem solving typically involves abductive reasoning.

We want to emphasize that our aim is not restricted to just developing computational systems, but rather importantly, to provide some insight into the cognitive issues related to principles of reasoning and problem solving. Thus, our strategy has been to do basic research in explicating the role of context in problem solving through examples from complex real-world problems. For this purpose we chose diagnostic problem solving, in particular medical problem solving as the pilot domain.

We identify two aspects of the notion of context as important: its role and its elements. We propose that a systematic investigation of the notion of context needs to be organised along these two dimensions. In section 2 we present a methodology for modelling context in diagnostic domains. Regarding the role that the context plays we distinguish between two basic issues. These are related to its roles in shifting the focus, and in capturing the focus of attention. These are taken up in section 3.2. Regarding the elements of context, which is closely related to the role it plays, our context-sensitive iterative diagnosis (ConSID) approach provides a high level context ontology which guides the process of explication and categorization of various types of context. This ontology is presented in section 3.3. In section 4 our ConSID approach is incorporated into the Creek architecture for knowledge-intensive case-based reasoning, while section 5 summarizes the paper.

2 A Task-Centered Methodology for Studying Context

This section summarizes the methodology that has been employed in this research, in the pursuit of modelling a context-sensitive diagnostic problem solving behaviour.

The process of modelling the contextual knowledge can be considered as consisting of two activities: (i) typing(categorizing) contextual knowledge, and (ii) identifying what kind of role each type plays, and where/when this happens during problem solving. Implicitly suggested in these two activities are the two perspectives that guide us toward an adequate account of the notion of context with respect to construction of knowledge-based systems. These perspectives are not mutually exclusive, but closely interrelated. They consider context, respectively as

- capturing a special type of knowledge and information, and
- a means for shifting attention of focus.

The first perspective invokes the study of how contextual knowledge can be categorized, and aims at constructing a context ontology that guides the modelling of contextual domain knowledge. The second perspective takes, as a starting point, an application task (in our case diagnosis) and identifies the subtasks of the overall task process on which some contextual influences are anticipated, as well as the role of contextual knowledge in each subtask. This perspective leads also to the study of the relationship between a task and the types of the context it relies on. We may say that the first perspective adopts a static view, while the second one is more concerned with dynamic aspects of context, i.e., its use.

Luc Steels explains components of expertise in terms of three concepts: task, domain and model (Steels, 1992). We follow his approach in our attempt to model diagnostic process at the knowledge level. Tasks correspond to the processes maintained in problem solving, such as diagnosis. Tasks are descriptions of what to do and why to do (i.e., the goal). Methods capture how- to-do knowledge. Tasks are decomposed into simpler subtasks iteratively until reaching undecomposable subtasks which are directly accomplishable.

A discipline that provided us much support and insight in developing a methodology for investigating context has been the medical research. The link between process and content, emphasized by Barrows (Barrows, 1994), related to teaching medical expertise guided us toward the way we study the contextual knowledge, a continuous attempt to keep the link between the reasoning process and the disease process.

The idea of the dependence between the knowledge and its use, proposed in AI by Chandrasekaran (Chandrasekaran, 1992), accords with Barrows' emphasis on the link between process and content in connection with problem-based curriculum in medicine.

According to van Heijst, Schreiber, and Wielinga (1997), the following activities were identified in the construction of a knowledge-based system: (i) construct a task model for the application; (ii) select and configure appropriate ontologies, and if necessary refine these; (iii) map the application ontology onto the knowledge roles in the task model; (iv) instantiate the application ontology with domain knowledge. Although we have a different motivation (than reusability which is one of the main motivations for van Heijst et. al.) for our study, we have found this account a useful basis for a knowledge level analysis of context. Our research strategy can be expressed in terms of the following research subtasks:

- construct a model of diagnostic 'process' knowledge. This serves to explicate all the involved application subtasks (i.e., diagnostic subprocesses),
- identify the **loci** of contextual effects (i.e., the subtasks where context may have an influence)
- identify the **role** context plays in each locus (i.e., the way it effects each subtask)
- identify the **source** of context that plays this role

As an example, Fig. 1 illustrates entities, such as loci and source, that our methodology explicates when applied to the medical domain.

Modelling the process of diagnosis involves identifying the subtasks which, when accomplished, lead to the achievement of the diagnostic goal, i.e., finding the fault causing the abnormal observations. It is exactly these subtasks that have a central role in modelling a shift of attention. This is because the subtasks are the *loci* where a shift in the focus of attention may be invoked. Knowledge level analysis starts out from different knowledge perspectives, e.g. a task perspective, a method perspective or a domain knowledge perspective. The tasks, in turn, identifies the need for knowledge, including contextual knowledge. We start by locating the subtasks of the whole diagnostic process where contextual knowledge is useful. After locating the points of contextual knowledge use, we try to understand the way these influences come into existence. The last step is the attempt to analyse the types of knowledge being utilized at these points.

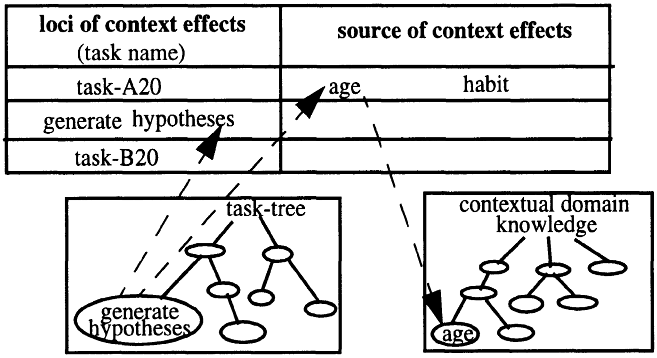


Fig. 1. The tasks in the task tree describe both the loci and source of contextual effects, in terms of the entities in the knowledge base

3 A Context-Sensitive Iterative Approach to Diagnostic Problem Solving

In domains where the knowledge is not represented by strong theories and the problems are to be solved with incomplete information, problem solving must be a context sensitive process. This section discusses a diagnostic problem solving model, which we call Context-sensitive iterative diagnosis (ConSID) approach which is particularly appropriate for open and weak domains. ConSID conceptualizes diagnosis as a context sensitive process integrating reasoning and action. The main principles on which the ConSID approach is based can be itemized as follows, where the last two points collectively reflect our account of the **role** of context in a diagnostic process.

- diagnosis is realized by an iterative application of abduction-prediction-induction cycle.
- the reasoning is goal-driven, and a shift in the focus of attention follows a change in the active goal.
- The focus of attention is captured in terms of core and contextual entities.

This approach to diagnosis is context-sensitive in two ways. First, the reasoning line is goal-driven (thus internal-context driven as explained in section 3.3). Second, it takes into consideration the contextual aspects of the grounded facts which allow the focus of attention to be captured in terms of both core domain and contextual concepts.

3.1 Iterative Application of Abduction-Prediction-Induction Cycle

A diagnostic label is an explanation of observed anomalies. Such a label is established by employing a strategy consisting of generation of tentative explanatory hypotheses and testing of these hypotheses on the basis of evidence which is incrementally gathered by performing actions in the physical world.

So, diagnosis is realized by a generate-and-test method which decomposes this task into two main components the first of which abductively generating hypothetical diagnoses, and the other testing these hypotheses collectively by employing predictive and inductive reasoning lines (see Fig. 2). This implies an iterative process because the hypotheses are generated with scarce information and through an abductive process which produces fallible results. As such, the hypotheses may render to fail explaining the newly gathered evidence. Consequently, a new hypotheses generation process may need to be invoked. This means a new cycle of hypotheses generation and of testing.

The generate-and-test method is utilized in two places in our model of diagnostic problem solving (see Fig. 2). These are when (i) generating and testing explanatory hypotheses (classification), and when (ii) generating and testing plans for gathering needed information (planning).

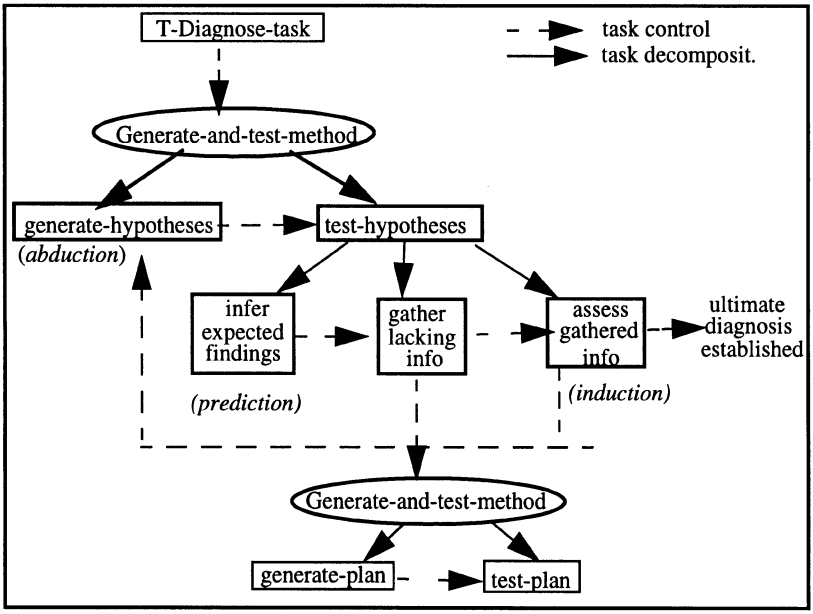


Fig. 2. The diagnostic task can be realized by an iterative application of abduction-prediction-induction cycle

3.2 The Role of Context

The role that context plays in a diagnostic process is twofold; as a means for shifting the focus, and as a means for capturing the focus.

We consider a diagnostic problem solving session as a sequence of task accomplishments. The order of the task execution determines the line of the problem solving. In our ConSID model, the goals of the reasoner which is a part of the internal context determines the task to be invoked. This happens through the link between the goals and the tasks; each goal is accompanied by a task. The relationships between the notions of goal, task, perspective and focus are summarized as follows:

goal → task → perspective → focus of attention → contextual and core domain entities

The focus of attention is switched when the goal of a reasoner changes. A change in the goal signals that a new task is to be performed. Attention reflects the epistemological needs of the reasoner when engaged in a dedicated task. Therefore, a task is the natural medium for discussing shifts in the focus of attention.

During the accomplishment of each task, the attention of the problem solver is focused on a small, relevant set of information and knowledge in the domain. Part of this focus consists of contextual elements, and the other part of core domain elements. We describe, for each task where the reasoner needs to focus her attention, through a mechanism we refer to as perspective. Thus, we employ a task-centered approach to the shift of focus where tasks are invoked by an internal context element, namely the goal, and the focus is captured by other contextual elements in combination with core domain elements.

3.3 The Elements of Context

The empirical work in cognitive psychology that studied context effects in some isolated tasks such as verbal learning and face recognition gives us insight into how to study the fault (e.g., disease) process and the context effects on that process. Various ways that the experiments are arranged in cognitive psychology gives us a possibility to interpret the differences between various types of contexts. Even though the choice as to the arrangement of experiments seems to be more or less arbitrary, or at most intuitive, findings are tangled, and the results are inconsistent and unsystematically presented, we found these empirical studies rather fertilizing for our construction of a general context ontology, illustrated in Fig. 3. In particular, the distinction between internal and external context has been influenced from two groups of experiments, one studying the state of the mind and the mood of the learner, and the other semantic properties of the target word to be learned as well as the environment in which learning takes place. This has also lead to our distinction between target-related and environment-related contextual elements.

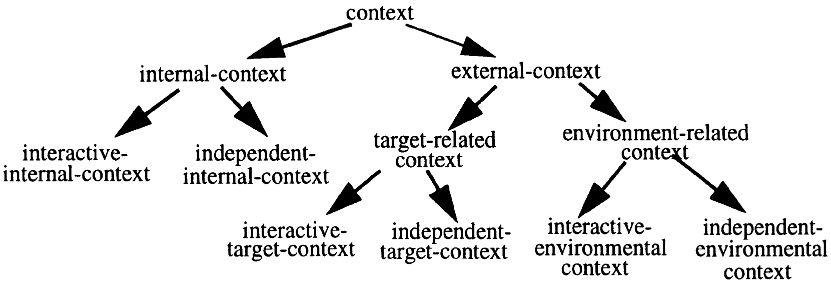

context
internal-context
external-context
interactive-internal-context
independent-internal-context
target-related context
environment-related context
interactive-target-context
independent-target-context
interactive-environmental context
independent-environmental context

Fig. 3. Context Ontology emphasizing the active role of the reasoner

The depth of processing theory (Craik and Tulving, 1975) has been another important inspiration source. This lead us to the distinction between interactive and independent types of context. Our interpretation of this theory is that we may include in a case the less processed contextual aspects of a phenomenon as independent context, while as the processing level gets deeper, the encoding becomes more interactive. As such, interactive type of context can be deeply represented and, thus, be relied on when explanation based reasoning is applied. The independent type of context, however, can only be represented in shallow level which does not allow its use in explanation based-reasoning. We will see in an implementation of ConSID approach the difference between the way in which independent and interactive type of context facilitates the reasoning process, or rather the type of reasoning paradigm.

4 ConSID-Creek System

ConSID-Creek is an implementation of the ConSID problem solving approach in medical domain, and has been build on the basis of Creek framework (Aamodt 1994a, 1994b).

The principle of encoding specificity (Tulving and Thompson, 1973) emphasizes the role of "experience" in learning and remembering. An experience is something more than, for example, a word to be learned and includes also the context in which the word is learned. The encoding specificity theory itself does not concentrate on elaboration of what this context consists of. The impact of encoding specificity principle on our research is that we emphasize the necessity of representing experiences. We conceptualized the representation of what is learned as an experience, that is, as a combination of context and core domain elements in form of 'cases'.

In our clinical diagnostic problem solving model, two types of cases are considered; explanation cases and plan cases. The first is the encoding of the explanation that connects the relevant features to the disease. The second is the encoding of a plan that includes a sequence of diagnostic and therapy actions, such as defining an examination protocol, selecting the way to perform a particular test, recommending a treatment, etc.

ConSID-Creek consists of a case based explainer and a case based planner, both of being the instances of case based reasoning paradigm. In this system new explanations are formed by retrieving and modifying previous explanations which have been useful in similar contexts. Similarly, new plans are made on the basis of "retrieve and modify" principle. We will not give in this paper the complete list of the *loci*, *role* and *sources* of contextual influences in a medical diagnosis process, as these have been identified and described in (Özturk and Aamodt, 1997). We identified there, in a systematic way, the relationships between contextual elements and the involved subprocesses, and explicitly constructed the corresponding link between these two. This made it possible to focus on the relevant context since it was explicated. As can be predicted, different contextual elements have been found relevant in case-based explanation and planning components of the system. Besides, different contextual elements have rendered to be relevant in various subprocesses of case based explanation and of case based planning processes.

In this paper, we concentrate on the role of context in one of the main subprocesses involved in medical diagnostic process, namely generation of hypotheses.

4.1 The Link Between Process and Content

The importance of the dependence between process and content has been emphasized in various disciplines (Barrows, 1994; van Heijst et al., 1997). In ConSID-Creek this link is provided by explicitly linking the tasks and the domain entities, through perspective description. The perspective is described in terms of both core domain and contextual domain entities.

A diagnostic *reasoning process* in the medical domain involves some tasks and some domain knowledge. The decision of which domain knowledge to include in a domain model can not be done in isolation from a model of *disease proces*s. Thus, the key to explicating the external context elements lies on a thorough model of disease process. In order to construct a context-sensitive model of the reasoning process, we need a context-sensitive model of the disease process.

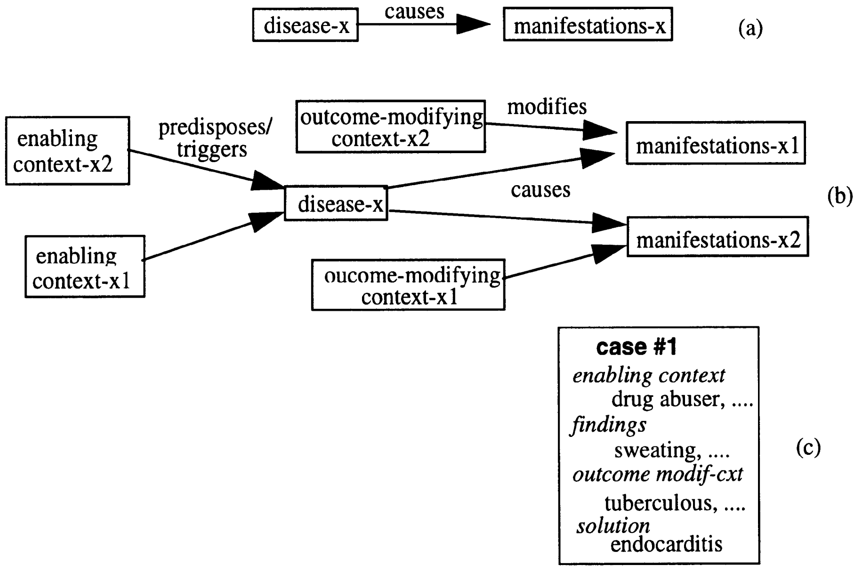


Fig. 4. (a) Traditional model of the disease process, (b) Our model of disease process, which allows multiple models for the same disease, (c) case structure

This is because, there will be correspondence between the domain concepts that is used in the model of the disease and the reasoning processes. Fig. 4 illustrates a traditional model of disease process (a) and our context-sensitive model (b). This model leads us towards a model of domain knowledge that embodies contextual aspects as well as core domain features.

The basic assumption underlying case based reasoning is that a past case is recalled on the basis of some criteria that renders it similar, in some respect, to the new case. Dreyfus and Dreyfus (Dreyfus and Dreyfus, 1986) draw attentions to the importance of an *overall similarity.* In our view, overall similarity emphasizes similarity of features within a certain context, not in isolation. That is, similarity of, for example, two cases can better be measured when the contexts of the cases are similar. Thus, contextual knowledge contributes to the improvement of similarity assessment.

Explanation cases in our system (see Fig. 4-c) embody various components of a disease process including enabling context, disease, outcome modifying context and manifestations. As such, this case model is able to represent the entire process, including all the related parts with regard to the issue of identifying a disease given a set of anomalous observations. So, this case representation is an excellent means for representing several possible versions of disease pictures, without leading to inconsistency in the knowledge base.

Our case content is similar, in the idea, to the illness scripts of Schmidt and Boshuizen (Schmidt and Boshuizen, 1993). We have augmented their cognitive model with *outcome-modifying context* (see Fig. 4-c) and used this model for the purpose of representing individual cases, instead of 'illness scripts'. The enabling context includes the factors that either predispose or trigger the onset of the disease, while the outcome modifying context influences the way a disease manifests itself.

Pieces of the enabling context as well as the outcome modifying context may be either independently or interactively encoded. The independent context is important with regard to shallow reasoning, for example, case based reasoning. Weak domains are typically lesser understood and, therefore, cannot always be deeply represented. In such domains the representation of individual experiences are rather important because, the knowledge which needs to be independently encoded may be included in case structures.

4.2 Context-sensitive Retrieval

The Creek framework is a conceptual model illustrating an integration of explanation-based and case-based reasoning paradigms.The central subtasks involved in Creek framework are retrieve, reuse, revise and retain (Aamodt, 1994a). In this paper we put a special emphasize on retrieval and illustrate how CBR-retrieval realizes the abductive subtask of generation of explanatory hypotheses.

The underlying case-based interpreter in Creek contains a three-step abductive process of 1) recalling relevant past cases 2) explaining derived consequences and new information within the activated knowledge structure, and 3) focusing towards a conclusion that conforms with the task goal. This "activate-explain-focus" cycle is a general mecha.nism that has been specialized for each of the four major reasoning tasks of the CBR cycle (Aamodt, 1994a).

In our system, the abductive subtask is realized by applying a case based reasoning method. The case based retrieval method decomposes the abductive task of generation of a hypotheses into subtasks which are realized by the primitive methods activate, explain and focus (see Fig. 5).

Activate: Activates a set of episodic memory structures, namely previously stored cases. It first establishes the context which serve as a means for pruning the number of activated cases. The context is established through the goal (i.e., recalling explanatory hypotheses) of the activation and by specifying the contextual elements that are relevant for that goal. Fig. 5 illustrates an example where the cases #113, #1 and #5 have been recalled after a superficial similarity assessment. This is a superficial similarity since it does not involve the deep knowledge. In this stage both independently and interactively encoded contextual features are used.

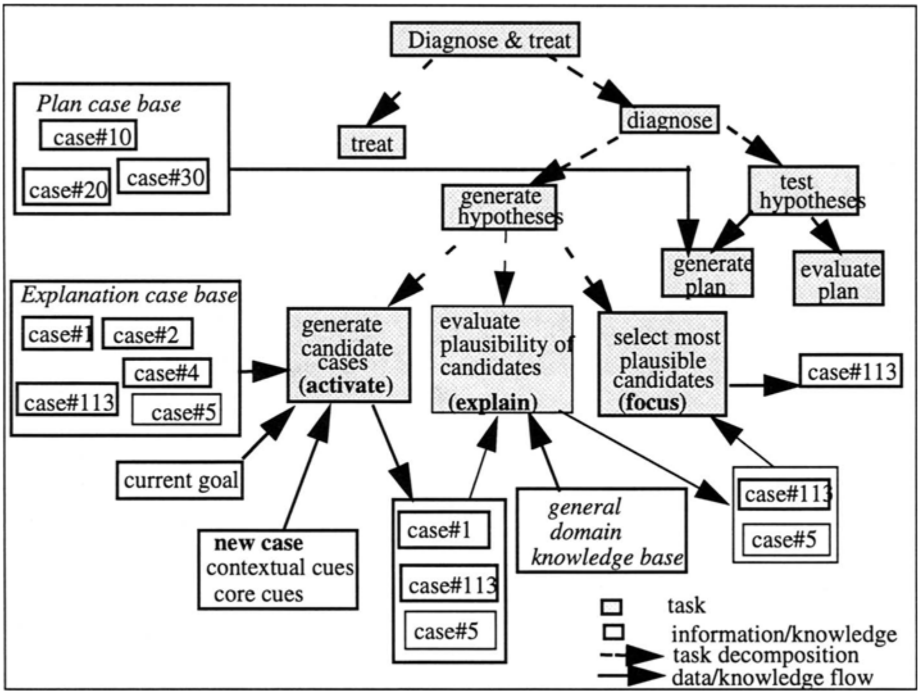
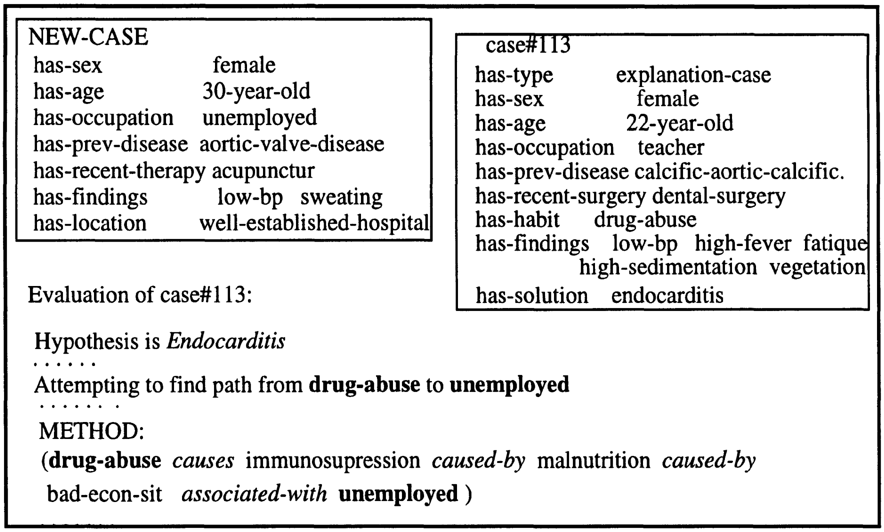


Fig. 5. Main subtasks of diagnostic problem solving and realization of these subtasks by CBR primitive methods activate, explain, and focus

Explain: Explains the similarity between the new case and each activated past case through employing deep knowledge. Fig. 6 illustrates how the deep knowledge is used in order to explain the deep similarities between two cases, namely the new case and the retrieved case, case#113. In this stage only interactively encoded feaures are used. In the activation stage, any similarity between 'unemployed' in the new case and 'drug abuse' in case#113 is not recognized. However, in the explanation stage, since it involves deep knowledge, a similaritty is detected. The details of this similarity is shown in Fig. 6. The explanation of the similarity: unemployed is associated with bad economical situation which may cause malnutrition which, in turn, may cause immunosupression. On the other side, drug abuse may also cause immunosuppression. So, 'drug abuse' and 'unemployed' may be considered similar as both may lead to immunosupression. This represents typically common sense reasoning.

Focus: Selects a small set of retrieved cases and uses their hypotheses as to-be-tested hypothesis, and thus for the guiding the determination of which further information is required for the diagnosis. Case#113 is selected as the working hypothesis in the example shown in Fig. 5.



NEW-CASE	
has-sex	female
has-age	30-year-old
has-occupation	unemployed
has-prev-disease	aortic-valve-disease
has-recent-therapy	acupunctur
has-findings	low-bp sweating
has-location	well-established-hospital

case#113	
has-type	explanation-case
has-sex	female
has-age	22-year-old
has-occupation	teacher
has-prev-disease	calcific-aortic-calcific.
has-recent-surgery	dental-surgery
has-habit	drug-abuse
has-findings	low-bp high-fever fatique high-sedimentation vegetation
has-solution	endocarditis

Evaluation of case#113:

Hypothesis is *Endocarditis*

. . . . . .

Attempting to find path from **drug-abuse** to **unemployed**

. . . .

METHOD:

(**drug-abuse** *causes* immunosupression *caused-by* malnutrition *caused-by* bad-econ-sit *associated-with* **unemployed**)

Fig. 6. A part of the explanation of similarities between the new case and the past case case#113

5 Conclusion

In this paper we have described a methodology that guides us when modelling the elements and the roles -influences- of the context in a diagnostic process. The methodology mediates a task-centered approach to conceptualizing a context-sensitive decision-support system. Decision-support systems that help solving problems in open and weak theory domains, i.e. hard problems, need improved methods to ground their models in real world situations. Models that attempt to capture domain knowledge in terms of, e.g. rules or deeper relational networks, tend either to become too abstract to be efficient, or too brittle to handle new problems. In our research we study how the incorporation of case-specific, episodic, knowledge enables such systems to become more robust and to adapt to a changing environment by continuously retaining new problem solving cases as they occur during normal system operation.

Case based reasoning paradigm relies on the notion of similarity as past cases similar to a new case are utilized in order to solve the new problem, instead of starting from the scratch. The use of experience has been stated in various disciplines as being a hallmark of expert behaviour (Patel & Groen, 1991; Dreyfus&Dreyfus 86). The similarity measure gains increasing importance for quality and efficiency of case based reasoning. As Dreyfus and Dreyfus emphasizes, a holistic template matching basing on an overall similarity gives much better results as to retrieving most useful past cases. We suggested in this paper that inclusion of contextual aspect to the measure of similarity is an important step toward realization of overall similarity.

6 References

Aamodt, A. (1994a). Explanation-driven case-based reasoning. In S. Wess, K. Althoff, M. Richter, Eds. *Topics in Case-based reasoning*, pp 274-288. Springer Verlag.

Aamodt, A. (1994b). A Knowledge Representation System for Integration of General and Case-Specific Knowledge. *Proceedings from IEEE TAI-94, International Conference on Tools with Artificial Intelligence.* New Orleans, November 5-12, pp 836-839.

Abu-Hakima, S. & Brezillon, P. (1995). Principles for the application of context in diagnostic problem solving.In Brezillon, P. and Abu-Hakima, Eds. *Working notes of the IJCAI-95 workshop on "Modelling context in knowledge representation and reasoning".*

Barrows, H. S. (1994) Practice-based learning: problem-based learning applied to medical education. Springfield.

Benjamins, R. (1993), Problem solving methods, dissertation, 1993.

Brezillon, P. (1993), Proceedings of the IJCAI-93 *Workshop on "Using knowledge in its Context"*, Laforia.

Chandrasekaran, B. (1992). Task-structure analysis for knowledge modeling. *Communications of the ACM*, Vol. 35, no. 9, Sep. 1992, pp 124-137.

Craik, F. I. M. and Tulving, E. (1975). Depth of processing and the retension of words in episodic memory. *Journal of experimental Psychology: general.* Vol. 104-3, pp. 268-294.

Dreyfus, H. L. and Dreyfus, S. E. (1986). Mind over machine. The free press. New York.

Patel, V. and Groen, G. J. (1991). The general and specific nature of medical expertise: a critical look, in Eds, Ericsson, K. A. and Smith, J. Towards a general theory of expertise. Cambridge university press, Cambridge, 1991

Schmidt, H. G. and Boshuizen, H. P. A. (1993). On acquiring expertise. *Educational psychology review*, vol. 5, No. 3, pp. 205-221.

Steels, L. (1990). Components of expertise. *AI Magazine.* Vol. 11. No 2. pp.30-49.

Tulving, E. and Thompson, D. M. (1973). Encoding specificity and retrieval processes in episodic memory. *Psychological Review.* Vol. 80. pp. 353-370.

van Heijst, G., Schreiber, A. Th., and Wielinga, B. J. (1997). Using explicit ontologies in KBS development. *International Journal of Human-Computer Studies.* Vol. 46, pp 183-292.

Özturk, P. and Aamodt, A.(1997) A context model for knowledge-intensive case-based reasoning. To be published in *International journal of human-computer interaction.*

A Framework for Developing Intelligent Tutoring Systems Incorporating Reusability

Eman El-Sheikh and Jon Sticklen

Intelligent Systems Lab, Computer Science Department, Michigan State University,
3115 Engineering Building, East Lansing, MI 48824 USA
{elsheikh, sticklen}@cps.msu.edu

Abstract. The need for effective tutoring and training is mounting, especially in industry and engineering fields, which demand the learning of complex tasks and knowledge. Intelligent tutoring systems are being employed for this purpose, thus creating a need for cost-effective means of developing tutoring systems. We discuss a novel approach to developing an Intelligent Tutoring System shell that can generate tutoring systems for a wide range of domains. Our focus is to develop an ITS shell framework for the class of Generic Task expert systems. We describe the development of an ITS for an existing expert system, which serves as an evaluation test-bed for our approach.

1 Introduction

The need for effective tutoring and training is rising, given the increasing complexity of the work place, and the knowledge-drain in contemporary commercial settings. This is especially true in industry and engineering fields. Rapid progress in science and technology has created a need for people who can solve complex problems and operate and maintain sophisticated equipment. Many Computer-Assisted Instruction (CAI) techniques exist that can present instruction, and interact with students in a tutor-like fashion, individually, or in small groups. The introduction of Artificial Intelligence techniques and Expert Systems technology to CAI systems gave rise to Intelligent Tutoring Systems (ITSs), i.e., intelligent tutors that can model the learner's understanding of a topic and adapt the instruction accordingly. Although ITS research has been carried out for over 15 years, few tutoring systems have made the transition to the commercial market. The main reasons for this failure to deliver are that the development of ITSs is difficult, time-consuming, and costly. There is a need for easier, more cost-effective means of developing tutoring systems.

In this paper, we describe a novel approach to developing an Intelligent Tutoring System shell that can generate tutoring systems for a wide range of domains. Our focus is to develop a family of ITS shells for the class of Generic Task (GT) expert systems. The goal is to develop an ITS architecture that can interact with any GT-based system, and produce an effective tutorial covering the domain knowledge represented in that system. As a test-bed for our approach, we describe the development of an ITS for the composite material design domain from a composite material design system that was built using the GT analysis and implementation methodology. An evaluation of the test-bed development project reveals the strengths and weaknesses of our approach.

The remainder of this paper describes the current state of our work on this topic, and constitutes a progress report. The next section highlights the problems with the traditional ITS approach and the motivation behind our approach. Section 3 details the background theories we build upon and describes related research and development efforts. In section 4, we present our approach for developing reusable ITSs. Finally, section 5 offers an evaluation of our results and a discussion of the contributions and future work.

2 Problem and Motivation

2.1 The Problem

A serious problem exists in the current methodology of developing Intelligent Tutoring Systems. Each application is developed independently, and tutoring expertise is hard-coded into individual applications. There is very little reuse of tutoring components, such as the student model, tutoring model, and user interface, between applications because we lack a standard language for representing the knowledge, a standard interface to allow applications to access the knowledge, and a set of tools to allow designers to manipulate the knowledge. In describing the state of building ITSs, Clancey and Joerger [4] lament that "...the reality today is that the endeavor is one that only experienced programmers (or experts trained to be programmers) can accomplish. Indeed, research of the past decade has only further increased our standards of knowledge representation desirable for teaching, while the tools for constructing such programs lag far behind or are not generally available."

2.2 Motivation for our Work

The motivation for our work comes from the need for reusable Intelligent Tutoring Systems and from the leverage that the Generic Task development methodology offers in solving this problem. The assumption of the GT approach is that there are basic "tasks" - problem solving strategies and corresponding knowledge representation templates - from which complex problem solving may be decomposed. Our goal is to develop an ITS architecture that can interact with any GT-based system, and produce an effective tutorial covering the domain knowledge represented in the problem solver. The backing intuition of this work is that GT systems are strongly committed to both a semantically meaningful knowledge representation method, and to a structured inferencing strategy, and by leveraging this strong structure, automated generation of tutorial overlays are enabled.

This approach facilitates the reuse of tutoring components for various domains. The ITS shell can be used in conjunction with any GT-based expert system, effectively allowing the same tutoring components to be plugged in with different domain knowledge bases [6]. As a test-bed for our approach, we are working on the development of an ITS in the industrial and engineering domain of composite material design and fabrication. Our goal is to develop an ITS for this domain utilizing only the domain knowledge represented within a GT-based composite material design system.

3 Background and Related Work

3.1 Expert System Development Methodology: Generic Tasks

The idea of Generic Tasks can be understood at one level as a semantically motivated approach to developing reusable software - in particular reusable shells for knowledge-based system analysis and implementation. Each GT is defined by a unique combination of: (1) a well-defined description of GT input and output form, (2) a description of the knowledge structure which must be followed for the GT, and (3) a description of the inference strategy utilized by the GT [3]. To develop a system following this approach, a knowledge engineer first performs a task decomposition of the problem, which proceeds until a sub-task matches an individual generic task, or another method (e.g., a numerical simulator) is identified to perform the sub-task. The knowledge engineer then implements the identified instances of atomic GT building blocks using off-the-shelf GT shells by obtaining the appropriate domain knowledge to fill in the identified GT knowledge structure. Having a pre-enumerated set of generic tasks and corresponding knowledge engineering shells from which to choose guides the knowledge engineer during the analysis phase of system development.

A number of these atomic Generic Tasks are currently available, and are implemented in our toolset. These include Structured Matching, Hierarchical Classification [8] and Routine Design [2]. The Knowledge Level Architecture (KLA) [16] provides an organizational overlay to the basic Generic Task approach to facilitate integration. Routine Design (RD) is the GT type we are interested in here, being the structure used to develop the test-bed expert system. RD was proposed as an architecture for performing design and planning tasks in which substantial experience is available (not for design or planning in totally novel situations). The generic task approach is only one approach for expert system development. For an overview on other task-specific approaches, see [18] or [15].

3.2 Test-Bed System

As a test-bed for our approach, we are working with a GT system called COMADE (COmposite MAterial DEsigner) [11], which is utilized in the design phase of composite material applications. The inputs to the system are characteristics of the target application. The output is a set of designs satisfying these requirements. Composites are more flexible than corresponding metals because there is a very wide range of choices for the various components of a composite. COMADE designs a composite material by finding satisfying choices for each of these components.

3.3 A Theory of Learning

To develop an effective tutoring system, we need to focus on the needs of the student, and how to address those needs using instructional techniques. This calls for an understanding of how the student uses and learns from the domain knowledge offered by the tutor. At the psychological level of analysis of learning, emphasis is placed on the construction of personal knowledge and on a learner-centered approach to pedagogy [13]. This approach to learning is commonly referred to as constructivism.

There is a distinction among the types of knowledge to be learned. Declarative knowledge involves knowledge of facts, concepts, and vocabulary. Procedural knowledge is demonstrated when a student can combine declarative knowledge so that it can be used in a course of action. Strategic knowledge involves knowing when and how to use declarative and procedural knowledge to construct a learning outcome. This type of knowledge epitomizes the active construction of knowledge and places the student at the center of the teaching-learning process.

Learning from strategic knowledge construction suggests that instruction for skill acquisition should, for the most part, be given in a problem-solving context [17]. This way, declarative and procedural knowledge can be converted into useful learning outcomes immediately. Learning is enhanced because the problem-solving context provides a set of conditions encoding the applicability of the knowledge and its relevance to problem-solving goals. This is the idea of learning by doing.

3.4 Related Work

This section describes the commonly-adopted standard architecture of an ITS and related work on the development of tutoring systems shells. There is widespread agreement within the ITS community that an ITS consists of four “expert” modules, as depicted in figure 1 [14]. The expert module contains the domain knowledge. The student model diagnoses what the student knows. The tutor identifies which deficiencies in knowledge to focus on and selects the appropriate instructional strategies to present that knowledge. The user interface is responsible for communication with the learner. This ITS framework will be extended in section 4.

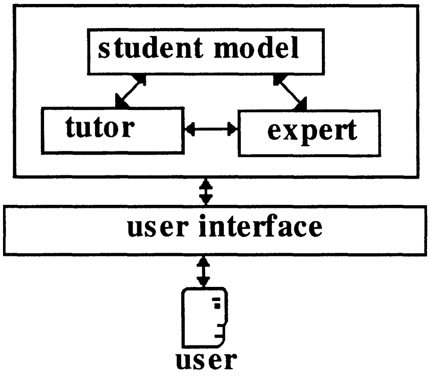


Fig. 1. Standard ITS architecture

Several other efforts are being made to develop ITS shells and reusable ITS components, although they are not discussed here due to space limitations. For more on these efforts, see [1, 7, 10, 12].

4 A Framework for Building Intelligent Tutoring Systems

4.1 Extending the Generic Task Framework

The goal of our work is to build an ITS framework that can interact with any GT-type problem solver to produce a tutoring system for the domain addressed by the problem solver. The learner interacts with both the tutoring system (to receive instruction, feedback, and guidance), and the expert system (to solve problems and look at examples), as shown in figure 2. Rather than re-implement the expert module for each domain, the ITS shell interfaces with a GT system to extract the necessary domain knowledge. This facilitates the reuse of the tutor, user interface, and student model components for different domains. Linking the ITS's expert module to the problem solver deserves special consideration. Rather than encode domain knowledge explicitly, the expert module extracts and tries to make sense of the domain knowledge available in the expert system. Thus, the quality of the tutoring knowledge is affected by the knowledge representation used by the expert system. The GT methodology's strong commitment to both a semantically meaningful knowledge representation method, and a structured inferencing strategy, allows the extraction of well-defined tutoring knowledge. The expert module can extract three types of knowledge: (a) decision-making knowledge (how the data relates to the knowledge), (b) knowledge of the elements in the domain knowledge base, and (c) knowledge of the problem solving strategy and control behavior.

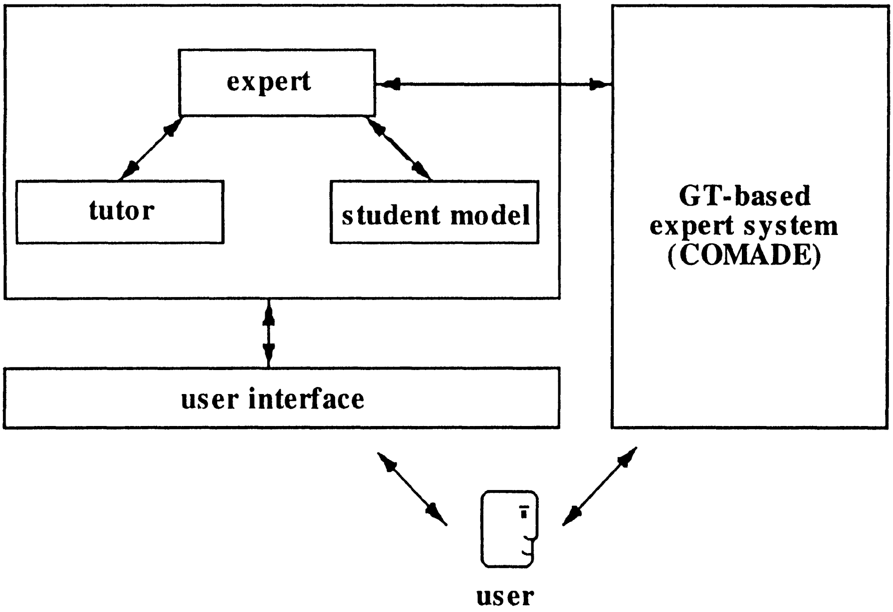


Fig. 2. System-user interaction model

To make the knowledge available to the ITS, the expert system must understand its own knowledge representation. The specific expert system used in this project, COMADE, is a Routine Design system, as described in section 3.1. We extend the structure of RD to allow any Routine Design system to have a self-understanding of its knowledge structures and reasoning processes. An RD system is composed of agents, each of which has a specific goal, purpose, and plan of action. For the extraction of tutoring knowledge, each agent must be able to answer three basic questions about itself:

1. WHAT did you do?
   The answer to this type of question is the result of the agent's action.
2. WHY did you do this?
   This is answered using a description of the agent's context (from its super-agent).
3. HOW did you do it?
   The answer is a description of the agent's actions, possibly including those of its sub-agent(s).

The expert module of the tutoring shell uses these answers, along with an encoding of a Routine Designer's structure to formulate domain knowledge as required by the ITS.

4.2 Completing the ITS Shell Framework

Thus far, we have explained the key component of the ITS shell that allows reuse, namely the expert module, and how it interfaces to a GT-based expert system. To complete the shell, this module must work in conjunction with a tutoring module and student model to interact with the learner. The main focus of this paper is to describe how reusability is achieved, but here we will briefly describe the other components.

Student modeling is the key to individualized knowledge-based instruction [9]. Within the context of the ITS, the student modeler plays an important role, in that it allows the tutoring system to adapt to the needs of individual learners. The approach we adopt for student modeling that seems to have potential is through the use of function-based reasoning. A functional model is used to simulate the learner and provide a description of the student state. The student's goals are modeled as a hierarchical model of functions. The top-level function or goal of the student is to learn the domain knowledge embedded within the problem solver. This main goal (function) is decomposed into lower level goals (functions) within the hierarchical model, which may be further decomposed into sub-goals (sub-functions) and so on. Figure 3 shows the Information Processing Task (IPT) Model for the student modeling component. The student model uses information provided from the user, expert module, and instructional module, to keep an accurate model of the learner's knowledge level and capabilities, and also to guide the instructional strategy. Our approach has several benefits: it provides the potential for generating explanations of learner behavior, knowledge, and misconceptions, as well as explanations of the reasoning process. In addition, a functional model can provide causal reasoning capabilities, and thus map onto human reasoning techniques.

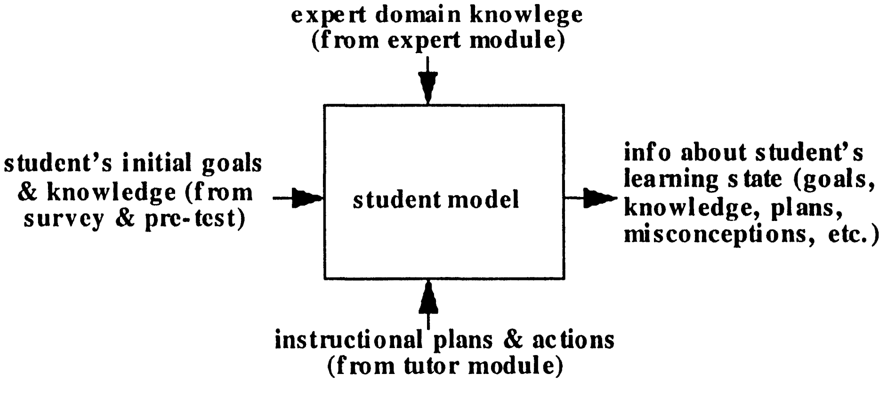


Fig. 3. IPT model for the student modeler

The tutor or instructional module uses two main instructional strategies, learning by doing and case-based teaching, in addition to question-answer templates. These teaching strategies are well-suited for teaching complex, knowledge-intensive domains, such as engineering domains. Moreover, they are a good match for our framework, since the learner can interact with both the expert system to solve problems and the tutoring system. Learning by doing is implemented within the shell by having the learner solve real problems using the expert system, with the tutor watching over as a guide. In the other learning mode, the tutor makes use of the case-based knowledge base of the expert system, in which the input-output sets are stored as individual cases. The instructional module can present prototypical cases to the user, which serve as a basis for learning from new situations. Alternatively, it can present a new situation, posed as a problem, and ask for a solution. The goal is to help the user develop a set of cases, and determine their appropriateness within the domain.

4.3 The COMADE Example

In this section, we present a description of how the user interacts with both the tutoring system and the expert system, a composite material designer called COMADE. There are two modes of interaction with the system. The first is tutor-driven and the instruction combines learning by doing and case-based teaching. In this mode, the tutor can either present and explain a case to the user, or ask the user to solve a case. This type of instruction is performed at two different levels of tutoring knowledge. At a low level, a case covers individual knowledge constructs (e.g., facts, concepts, processes). This allows the tutoring of declarative and procedural knowledge. At a higher level, a case consists of a complete knowledge set to solve the top-level problem addressed by the expert system. In the case of COMADE, a high-level case would include all the knowledge and reasoning required to design a composite material satisfying certain input requirements. This type of case facilitates strategic knowledge learning by the user. Example interactions are shown in figures 4 and 5.

```
Tutor: To develop a composite material application with
properties: required flexural modulus = 40 GPa,
required tensile strength = 2000 MPa, flame retardance
= no,... {other requirements}, we can use the following
material options: use Polyamide-66 as the matrix type,
P100 graphite as the fiber type, a continuous fiber
length, and no chemical agent. This choice was made
because...{explanation of the reasoning process for the
solution path}.
```

Fig. 4. Example of tutor-driven interaction: presenting a high-level case

```
Tutor: Given the following design requirements:
required flexural modulus is 420 GPa, and required
tensile strength is 1390 MPa, what type of fiber is
most appropriate?

User: P-100 Graphite seems the most appropriate.

Tutor: Correct.
```

Fig. 5. Example of tutor-driven interaction: solving a low-level case

The second mode of system interaction is user-driven, in the form of question-answer sessions. In this mode, the user chooses what he/she wants to learn about, by asking the system questions. The tutor can answer three types of questions: (1) decision-making questions, (2) questions about the elements in the domain knowledge base, and (3) questions about the problem solving strategy and control behavior. Figure 6 shows examples of question-answer sessions for the three types of questions.

User: Why did you choose a glass material as the fiber reinforcement? (Type 1)

Tutor: I chose E-glass because the required flexural modulus is 40 GPa, the required tensile strength is 2000 MPa, and the chemical environment is not an acid. E-glass is an appropriate fiber for these requirements.

User: What is the appropriate range of use temperature for a Polyamide-66 material? (Type 2)

Tutor: The appropriate use temperature for Polyamide-66 is between 120 - 140 C.

User: What does the reinforcement plan do? (Type 3)

Tutor: In the context of determining an appropriate reinforcement for the composite material, the reinforcement plan selects an appropriate fiber material and length.

Fig. 6. Example question-answer sessions

5 Results and Discussion

We have presented the framework of a shell for building Intelligent Tutoring Systems for various domains by interfacing with existing expert systems, and reusing the other tutoring components. Specifically, we have described the development of an ITS for the composite materials design domain. More generally, we have formulated a technique for leveraging the knowledge representation and structure of the Routine Design framework for tutoring. And we have laid the foundation of our top-level goal - to develop a framework for an ITS extension to the GT approach. The main problem we addressed is reusability. We have found a solution to this problem in the development of tutoring systems for existing expert systems and knowledge bases.

Further research is needed to make the framework more concrete. Future research paths include developing an ontology of the tutoring knowledge available within a GT-based expert system. We also hope to test our approach further by applying it to other existing expert systems. Testing our ITS shell using several expert systems will provide more insight into the benefits and limitations of our approach.

References

1. Antao, B., Broderson, A., Bourne, J., and Cantwell, J.: Building intelligent tutorial systems for teaching simulation in engineering education. *IEEE Transactions on Education*, 35(1), pp. 50-56 (1992)
2. Brown, D.: Design. *Encyclopedia of AI*, 2nd edition (S. Shapiro, ed.), Wiley-Interscience Publishers, New York, NY (1991)
3. Chandrasekaran, B.: Generic tasks in knowledge-based reasoning: high-level building blocks for expert system design. *IEEE Expert*, 1(3), pp. 23-30 (1986)
4. Clancey, W., and Joerger, K.: A practical authoring shell for apprenticeship learning. *Proc. of ITS'88: First Intl. Conf. on Intelligent Tutoring Systems*, Montreal, Canada, pp. 67-74 (1988)
5. Costa, E.: *New Directions for Intelligent Tutoring Systems*, Springer-Verlag Publishers, New York, New York (1991)
6. El-Sheikh, E., Kamel, A., and Sticklen, J.: An ITS shell leveraging the generic task approach to problem solving. *Proc. of the ITS'96 Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs*, Montreal, Canada (1996)
7. Fleming, J., and Horwitz, C.: Applications of the rapid intelligent tutoring system development shell. *Proc. of the ITS'96 Workshop on Architectures and Methods for Designing Cost-Effective and Reusable ITSs*, Montreal, Canada (1996)
8. Gomez, F., and Chandrasekaran, B.: Knowledge organization and distribution for medical diagnosis. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11(1), pp. 34-42 (1981)
9. Holt, P., Dubs, S., Jones, M., and Greer, J.: The state of student modelling. In: *Student Modelling: The Key to Individualized Knowledge-Based Instruction* (J. Greer and G. McCalla, eds.), Springer-Verlag Publishers, Berlin, Germany, pp. 3-35 (1994)
10. Kameas, A.: The functional architecture and interaction model of a generator of intelligent tutoring applications. *Journal of Systems Software*, 36, pp. 233-245 (1997)
11. Lenz, T., McDowell, J., Moy, B., Sticklen, J., and Hawley M.: Intelligent decision support for polymer composite material design in an integrated design environment. *Proc. of the American Society of Composites 9th Technical Conference*, pp. 685-691 (1994)
12. Murray, T.: Having it all, maybe: design tradeoffs in ITS authoring tools. *Proc. of ITS'96: Third Intl. Conf. on Intelligent Tutoring Systems*, Montreal, Canada, pp. 93-101 (1996)
13. Phye, G.: Learning and remembering: the basis for personal knowledge construction. In: *Handbook of Academic Learning: Construction of Knowledge* (Gary Phye, ed.), Academic Press, San Diego, California, pp. 47-64 (1997)
14. Polson, M. C., and Richardson, J. J.: *Foundations of Intelligent Tutoring Systems*, Lawrence Erlbaum Publishers, Hillsdale, New Jersey (1988)
15. Steels, L.: Components of expertise, *AI Magazine*, 11(2), pp. 28-49 (1990)
16. Sticklen, J.: Problem solving architecture at the knowledge level. *Journal of Theoretical and Applied Artificial Intelligence*, 1, pp. 1-52 (1989)
17. Wenger, E.: *Artificial Intelligence and Tutoring Systems: Computational and Cognitive Approaches to the Communication of Knowledge*, Morgan Kaufmann Publishers, Los Altos, California (1987)
18. Wielinga, B.: Schreiber, A., and Breuker, J., KADS: A modeling approach to knowledge engineering. *Knowledge Acquisition*, 4(1), pp. 5-54 (1992)

The Classification and Specification of a Domain Independent Agent Architecture

A. Goel and K.S. Barber

The Laboratory for Intelligent Processes and Systems
The Department of Electrical and Computer Engineering, ENS 240
The University of Texas at Austin
Austin, TX 78712-1084, USA
e-mail: barber@mail.utexas.edu

Abstract. The use of multi-agent systems to model and control complex domains is increasing. The application of multi-agent systems to distributed real-world domains motivates the development of domain independent agent architectures to foster the re-use of existing research. A domain independent agent architecture is defined as a set of representations, reasoning mechanisms and algorithms that can be used to define the behavior of an agent. Ideally, a classification method for agent architectures should ease the process of requirements matching between domains and architectures by allowing system designers to understand the relationship between domains and architectures. This paper presents the need for a classification schema and proposes a starting point for future discussion. The Sensible Agent architecture is then introduced within the classification schema. Sensible Agents perceive, process, and respond based on an understanding of both local and system goals. The core capabilities and internal architecture of a Sensible Agent is given as an example of how accurate specifications can help foster the re-use of research.

Introduction

Multi-agent systems offer advantages over traditional, centralized systems when modeling and controlling large, complex systems by distributing core capabilities and modeling the large number of possible interactions without explicit representation. In order to leverage the advantages of multi-agent systems, it is necessary to characterize domains and model agent architectures in a common language that allows for system designers to match the needs of a domain with the capabilities of an architecture. This paper covers the need for a common model representation language or an architecture classification method and how the Sensible Agent architecture has been defined to achieve these goals.

Sensible Agents are essentially goal driven entities that function inside of a multi-agent system. The term "sensible" derives from the agents' awareness of the relationship between their own goals and a set of system level goals. A Sensible

Agent: ***i)*** maintains a representation of both behavioral and declarative knowledge of itself and other system agents, ***ii)*** processes or develops an understanding of both local (i.e. its own) goals and system goals, prioritizes these goals and drives the agent behavior according to its behavioral and declarative knowledge, and ***iii)*** performs actions with respect to its own internal events or external events from other system agents or the environment [1]. The internal representations and reasoning strategies of Sensible Agents are defined in a domain-independent that supports the inclusion of domain-specific information. This allows Sensible Agents to be deployed across a variety of domains with a minimum of architectural effort.

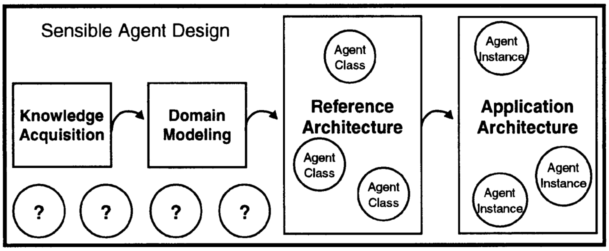


Fig. 1. Agent Design Methodology

The typical development process for agent architectures consists of four major tasks, as shown in Fig. 1 [11]:

- *knowledge acquisition*—domain knowledge is acquired from experts and formally documented.
- *domain modeling*—knowledge is synthesized from various sources, verified by the original experts and then rendered as models that characterize the domain requirements.
- *reference architecture*—the resulting domain models are then formally partitioned into the responsibilities and services provided by different agent classes in the system by either: ***i)*** defining a new architecture or ***ii)*** using portions or all of a pre-existing architecture.
- *application architecture*—agents within the reference architecture are then implemented.

In existing multi-agent research, focus has been placed on defining new architectures for problem-specific solutions, rather than developing stable, domain-independent architectures that can be adapted for new domains. Choosing from pre-existing architectures requires a clear understanding of domain requirements and specifications (*needs*) and agent architecture capabilities and assumptions (*amenities or constraints*). Hayes-Roth discusses the relationship between domains and architectures [9],

> "...to a large degree, an agent's architecture determines its potential behavior and, therefore, the niches in which it potentially can function: Agent Architecture→Potential Behavior→Suitable Niches... to function effectively in a particular niche, an agent must exhibit the range of behavior required in that niche and, therefore, must have an architecture that supports the required behavior: Intended Niche→Required Behavior→Sufficient Architectures."

For example, the following agent architectures and testbeds stand out as having provided key insight into multi-agent research: ARCHON [17], IMAGINE [14], DESIRE [3], DVMT/MACE [2], SOAR [13] and AIS niches [9]. Unfortunately, the lack of a standard classification schema prevents designers from understanding how these agent architectures are similar or different. Without such a classification schema, formalizing domain needs and matching them with architecture capabilities is a difficult task. Exacerbating the problem, many agent architectures do not: ***i)*** provide their minimum requirements for agency, ***ii)*** distinguish between an agent's domain-specific and domain-independent services, and ***iii)*** specify the process of populating their internal structures with domain-specific information.

The next section provides a short description of one classification method. Sensible Agents are used as examples of how an architecture may be defined relative to the classification. Section 0 presents the Sensible Agent Architecture as a domain-independent architecture with the underlying representations and reasoning mechanisms that support domain-specific information.

The Classification of Agent Architectures

Classification of multi-agent architectures can provide future system designers with the information required to select among different architectures and implementation platforms. This classification developed in this research can be used to model domain requirements with respect to multi-agent systems; providing a method for requirements matching. The classification method is defined along three axes: ***i)*** agent behavioral style, ***ii)*** communication medium, and ***iii)*** interaction styles.

Agent behavior can be classified by the degree of cooperation and level of self-preservation an agent exhibits [16]. The behavioral styles will shape an agent's choice for execution. As such, the behavioral style is a by-product of the *motivations* of an agent resulting in the selection of one course of action over another.

In Sensible Agents, these motivations are embodied in goals assigned to each agent. Agents are motivated to accomplish goals by expecting some utility for their successful completion. Since Sensible Agents operate with possibly distinctive (e.g. subjective) viewpoints, the notion of utility is separated into agent utility and system utility [6]. This dual notion of utility supports the placement of an agent along the full spectrum of behavioral style listed earlier (cooperation and self-preservation) [16] [6].

The type of communication channels open to an agent are vital in governing the styles of interaction between agents. All multi-agent architectures make some implicit assumptions concerning the availability or style of communication available to agents. Specifically, research addressing communication problems and solutions is relevant to multi-agent research since the resulting investigations target the reasoning process of these systems in response to the information exchanged. To classify the architecture, it is necessary to accurately characterize both the types of tolerated failures, and the level of connectivity among agents that is required by the domain or assumed by the architecture. We have developed the following classifications:

- *Flawless and Ubiquitous*—Agents may not fail, can perform all tasks requested of them and ubiquitous communication exists.
- *Fail-Stop*—Agents may fail during operation but failure is recognized by all agents in the system. Ubiquitous communication still exists.
- *Restricted Communication*—Agents may not fail, can perform all tasks requested of them and communication is restricted. Communication restrictions are known to all agents.
- *Crash Failure*—Agents may fail during operation but the failure is not necessarily recognized by other agents. Communication restrictions are known to all agents.
- *Crash Communication*—Agent may not fail and can perform all tasks but communication restrictions are not known to all agents.
- *Full Uncertainty*—Agents may fail during operation and communication restrictions are not known to all agents.
- *Nil Communication*—No direct communication exists between agents.

Sensible Agents do not make any assumptions regarding the level of communication that is available to all agents. Sensible Agents can operate effectively in environments up to *Full Uncertainty* but their use in *Flawless* or *Nil Communication* domains may not be warranted. Sensible Agents have some internal reasoning mechanisms supporting their operation in *Full Uncertainty* environments that may introduce unnecessary overhead.

The types of interactions available to agents can also affect the types of solutions agents reach and which environments a system of agents may effectively operate in. In deciding the appropriate interaction types for any given situation, agents are implicitly altering the manner in which they plan for goals or execute actions. This research classifies a system's handling of interaction styles as: ***i)*** Unstructured, ***ii)*** Static, and ***iii)*** Dynamic.

Unstructured architectures or systems do not have any rules or conventions governing how agents interact with one another. Agents may freely pass goals or information to each other without regarding the utility of such actions for themselves or the system as a whole. IMAGINE [14] and ARCHON [17] are examples of systems where agents can send requests among themselves but no structures exist for evaluating the utility or worth of satisfying these requests.

Static architectures use explicitly defined interaction styles that do not change in response to environmental state. The hierarchy of information and authority among agents is pre-defined. In a coordinator/cohort relationship, the cohort will always carry out the instructions provided by the coordinator, regardless of their impact. It has been noted by Glance that the level of cooperation in a static group is extended beyond that of an unstructured group yet the inability to dynamically reorganize remains a limiting factor [4]. Additionally, using a static interaction style assumes that the optimal configuration can be pre-determined and will not change with time, a condition that can not be guaranteed.

Dynamic architectures are able to alter their interaction styles in response to changing environmental conditions in order to increase the quality of the solution provided. Such an assumption may be necessary in a dynamic, uncertain environment where the optimal interaction style is neither constant nor predictable. The ability to shift positions in an organizational structure (thereby altering the interaction style) allows agents to explore new avenues of cooperation.

Characterizing an architecture according to its behavioral style, communication medium, and interaction style provides a starting point from which to begin the task of architecture classification. This classification provides a high-level method for comparing different architectures to judge their suitability for different domains.

Sensible Agent Architecture

The development of flexible, responsive and adaptive systems is a highly desirable goal when deploying automated systems operating in complex and dynamic domains. The Sensible Agent architecture is proposed with this goal in mind. Capable of dynamically adapting their level of autonomy, Sensible Agents perceive, process and respond based on an understanding of both system goals and their own local goals. We provide an overview of the Sensible Agent architecture and discusses the methodologies by which domain-specific responsibilities are assigned to the Sensible Agent's domain-independent representations.

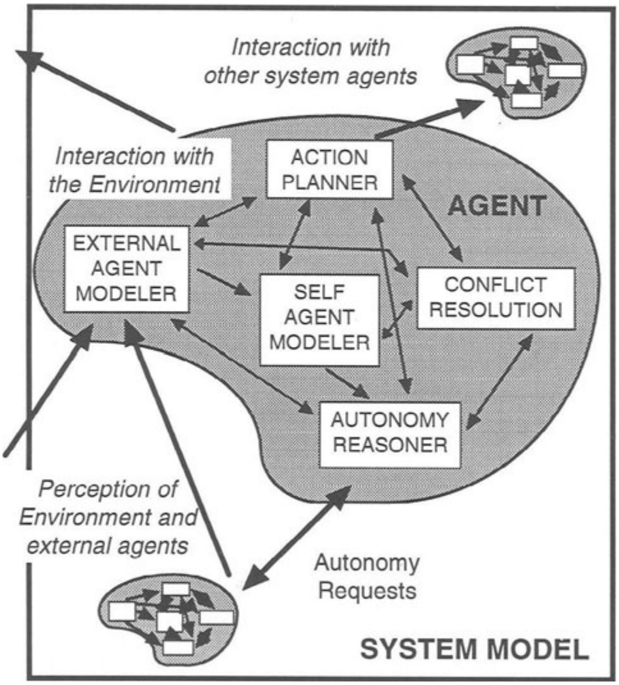


Fig. 2. Sensible Agent Architecture

The internal structure of a Sensible Agent is shown in Fig 2., and is composed of five modules [1]:

- A *self-agent modeler (SAM)* contains the behavioral model of the agent [7]. This module interprets internal or external events acting on the agent and changes its state accordingly. Other modules access this model for necessary information.
- An *external agent modeler (EAM)* contains knowledge about other agents and the environment [7]. This module maintains beliefs about states and events external to the agent and predicts the actions of other agents. Other modules within the self-agent can monitor this model for changes external to the self agent that affect their reasoning processes. Both the SAM and EAM use Extended State Charts (ESCs), which are based on Statecharts [8] with respect to temporal and hierarchical extensions [15].
- An *action planner (AP)* solves domain problems, stores agent goals, and executes problem solutions. As such, the AP is responsible for generating plans and choosing an execution path to support those plans. Of all the modules, the AP contains the most domain-specific knowledge since it is heavily involved with the planning process.
- A *conflict resolution advisor (CRA)* identifies, classifies, and generates possible solutions for conflicts occurring between the self-agent and other agents [10]. Selection of the appropriate resolution strategies used (e.g. evade, ignore, re-plan or acquiesce) are based both on the autonomy levels of the goals in conflict (see autonomy reasoner below) and the underlying reason for the conflict (e.g. goal, plan or belief). This module monitors the AP, SAM, and EAM to identify conflicts then offers conflict resolution strategies to the AP or AR.
- An *autonomy reasoner (AR)* determines the appropriate autonomy level for each goal an agent possesses, assigns an autonomy level to each goal, and reports autonomy-level constraints to other modules. It also handles all autonomy level transitions and requests for transitions made by other agents. A critical consideration for Sensible Agent behavior is the agent's level of autonomy. The term *level of autonomy* refers to the types of roles an agent plays in its planning interactions with other agents and is based on four constructs [12]:
- *Responsibility* is a measure of how much the agent must plan to see a goal solved.
- *Commitment* is a measure of the extent to which a goal must be solved.
- *Authority* is a measure of the agent's ability to access system resources.
- *Independence* is a measure of how freely the agent can plan.

These four autonomy levels provide a guide for agent planning, ensure consistent agent behavior within autonomy levels, enable agents to choose productive problem-solving groups, and support flexible responses to constraints.

The utility of Sensible Agents derives from their flexible interaction styles in dynamic, uncertain environments and their modular architecture. We support third party usage of the architecture by providing: ***i)*** clear definitions of our requirements for agency, and ***ii)*** public interfaces for the different modules in a Sensible Agent.

The modules within the Sensible Agent Architecture ensure that core agent competencies are met. The following is a list of the core capabilities required for agency in the Sensible Agent Architecture. These requirements can help system designers decide if this architecture is appropriate for their domains. Specifically, an agent must [5]:

- *have an encapsulated set of capabilities that it can perform independently of the other agents in the system*
- *have a local view of the system in which it is working*
- *have the facilities (and motivation) to communicate with other agents*
- *have both system and local goals in needs to accomplish*
- *be endowed with a set of motivations to drive its behavior*
- *derive some utility from the ability to change autonomy levels for a respective goal*

A systems engineering process is underway to specify a set of public interfaces for each of the five modules discussed earlier. Well-defined interfaces and the allocation of responsibility among modules insure a formal specification for integration requirements as well as a specification of capabilities. Public interfaces allow the module implementations to be performed independently of one another. As long as an implementation adheres to its public interface, it will be able to function inside of a Sensible Agent. Additionally, new research can be performed by plugging in new implementations or algorithms into a pre-existing structure, thus fostering re-use of previous efforts as well as rapid evaluation of new techniques for delivering agent services.

Summary

The proposed multi-agent architecture classification provides system designers with a tool by which to evaluate the correlation of domain requirements to architecture capabilities. Additionally, the classification method can provide a method for comparing and contrasting different agent architectures. It is important to note that the classification method does not support the objective ranking of different agent architectures, only their ranking relative to the needs of a specific domain.

The Sensible Agent Architecture has been presented within the context of classification categories: behavioral style, communication medium, and interaction styles. The minimum requirements for agency and the importance of well-defined public interfaces for internal modules in Sensible Agents have also been presented. By defining the requirements for agency, future designers can decide if Sensible Agents offer the required services for their own domains. The public interfaces allow for new research to occur while leveraging prior architectural and implementation decisions. The architecture provides a foundation or blueprint of the agent capabilities; promoting the ability to rapidly insert new algorithms or representations as the technology evolves.

Similar specifications for other multi-agent architectures may well foster: ***i)*** an improvement in the process of requirements matching between domains and architectures, ***ii)*** the re-use of existing architectures, and ***iii)*** the exchange of modules or algorithms among researchers investigating multi-agent systems.

Bibliography

[1] K. S. Barber. (1996) “The Architecture for Sensible Agents,” , In Proceedings of the *International Multidisciplinary Conference, Intelligent Systems: A Semiotic Perspective*, Gaithersburg, MD, October 20-23, 1996.

[2] A. H. Bond and L. Gasser. (1988) “An Analysis of Problems and Research in DAI,” in *Readings in Distributed Artificial Intelligence*, A. H. Bond, Ed. San Mateo, CA: Morgan Kaufmann Publishers Inc., 3-35.

[3] F. Brazier, B. D. Keplicz, N. R. Jennings, and J. Treur. (1995) “Formal Specifications of Multi-Agent Systems: a Real-World Case,” pp. 25-32, In Proceedings of the *First International Conference on Multi-Agent Systems*, San Francisco, California, June 12-14, 1995.

[4] N. S. Glance and B. A. Huberman. (1993) “Organizational Fluidity and Sustainable Cooperation,” pp. 89-103, In Proceedings of the *5th Annual Workshop on Modelling Autonomous Agents in a Multi-Agent World*, Neuchatel, Switzerland, August 25-27, 1993.

[5] A. Goel. (1997) "A Formal Specification and Classification Schema for Agent Architectures," Master's Thesis, Department of Electrical and Computer Engineering, The University of Texas at Austin, Austin, TX.

[6] A. Goel, T. H. Liu, and K. S. Barber. (1996) “Conflict Resolution in Sensible Agents,” pp. 80-85, In Proceedings of the *International Multidisciplinary Conference on Intelligent Systems: A Semiotic Perspective*, Gaithersburg, MD, October 20-23, 1996.

[7] T. Graser and K. S. Barber. (1996) “Meta-Modeling of Sensible Agents,” , In Proceedings of the *International Multidisciplinary Conference on Intelligent Systems: A Semiotic Perspective*, Gaithersburg, MD, October 20-23, 1996.

[8] D. Harel. (1987) “Statecharts: A Visual Formalism for Complex Systems,” *Science of Computer Programming* 8: 231-274.

[9] B. Hayes-Roth. (1995) “An Architecture for Adaptive Intelligent Systems,” *Artificial Intelligence* 72: 331-365.

[10] T.-H. Liu, C. Chuter, and K. S. Barber. (1997) “Virtual Environment Simulation for Visualizing Conflict Resolution Strategies in Multiple Robot Systems,” pp. 154-158, In Proceedings of the *5th IASTED International Conference, Robotics and Manufacturing*, Cancun, Mexico, May 28-31, 1997.

[11] C. E. Martin. (1997) "Representing Autonomy in Sensible Agent-based Systems," Master's Thesis, Department of Electrical and Computer Engineering, University of Texas at Austin, Austin, TX.

[12] C. E. Martin and K. S. Barber. (1996) “Representation of Autonomy in Distributed Agent-based Systems,” , In Proceedings of the *International Multidisciplinary Conference, Intelligent Systems: A Semiotic Perspective*, Gaithersburg, MD, October 20-23, 1996.

[13] A. Newell. (1990) *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

[14] D. D. Steiner. (1996) “IMAGINE: An Integrated Environment for Constructing Distributed Artificial Intelligence Systems,” in *Foundations of Distributed Artificial Intelligence*, G. M. P. O'Hare and N. R. Jennigns, Eds. New York: John Wiley and Sons, Inc., 345-364.

[15] A. Suraj, S. Ramaswamy, and K. S. Barber. (1996) “Extended Statecharts: A Specification Formalism for High Level Design.,” , In Proceedings of the *Fourth International Conference on Control, Automation, Robotics, and Vision (ICARCV)*, Singapore, December 3-6, 1996.

[16] E. Werner. (1991) "The Design of Multi-Agent Systems," pp. 3-28, In Proceedings of the *Proceedings of the 3rd European workshop on Modelling Autonomous Agents in a Multi-Agent World*, Kaiserslautern, Germany, August 5-7, 1991.

[17] T. Wittig. (1992) "ARCHON: An Architecture for Multi-Agent Systems," . New York: Ellis Horwood, pp. 135.

MAVE:
A Multi-agent Architecture for Virtual Environments[1]

Jeffrey Coble Karan Harbison

The University of Texas at Arlington
Center For advanced Engineering and Systems Automation Research (CAESAR)
500 W. 1st St., MS #19066 Rm. 412 Woolf Hall
Arlington TX, 76019
817-272-3321
{coble,harbison}@caesar.org

Abstract. This paper describes a Multi-agent Architecture for Virtual Environments (MAVE). The foundation of this research is the development of a two-tier architecture. The first tier of the architecture is an object oriented physical representation of the virtual environment that is designed to mimic the logical decomposition. The second tier of the architecture is designed to support the need for persistence, real-time interface to external data sources, distribution, and collaboration. MAVE addresses the need for autonomous components that support re-use, access to component level services, and intelligent behaviors.

1 Virtual Environments

One of the most interesting new technologies is the 3D distributed virtual environment. Distributed virtual environments are computer-generated, three dimensional, interactive visualizations, which unlike simulations, need only suggest a real or imagined space[1]. They are not necessarily photo-realistic nor do they provide a sense of total immersion. What they do provide is a very compelling, multi-user environment in which data objects, resources, conversations, and applications can be shared among participants. They can also provide a much more intuitive interface to complex information by capitalizing on human perceptual processes and intuitive human experiences.

We have applied the concept of a virtual environment to several different domains. Some of the more appealing applications are providing interfaces to the control of complex manufacturing processes such as job shop scheduling systems or logistical planning systems. For this paper, we will discuss our research in the context of a

[1] This work was supported by DARPA under contract N66001-95-D-8656 and MDA972-94-2-0010 and by NIMA under grant NMA202-97-1-1022

logistical planning system interface. Our work is focused on providing a revolutionary method of improving the productivity and usefulness of complex systems.

2 Scenario

Achieving efficient, real-time control of a logistics pipeline is dependent upon the ability of users to sensibly navigate and manipulate the large amounts of data associated with plan-related activity. Such activities include the visualization of complex planning data, the formulation, refinement, and submission of planning tasks to planning systems, and the ability to collaborate and disseminate such information among end users and plan managers.

Consider this vision of a virtual environment in which exists a 3D, Virtual Reality (VR) visualization of both the plan hierarchy and the ongoing steps of the real execution (Figure 1): This web-based VR representation allows users to 'walk' through the plan and execution models, zoom in on areas of interest and view the plan/execution nodes from a variety of hierarchical levels. Each facet of the nodes exploits different pieces and representations of the available information and is updated in real time from external data sources.

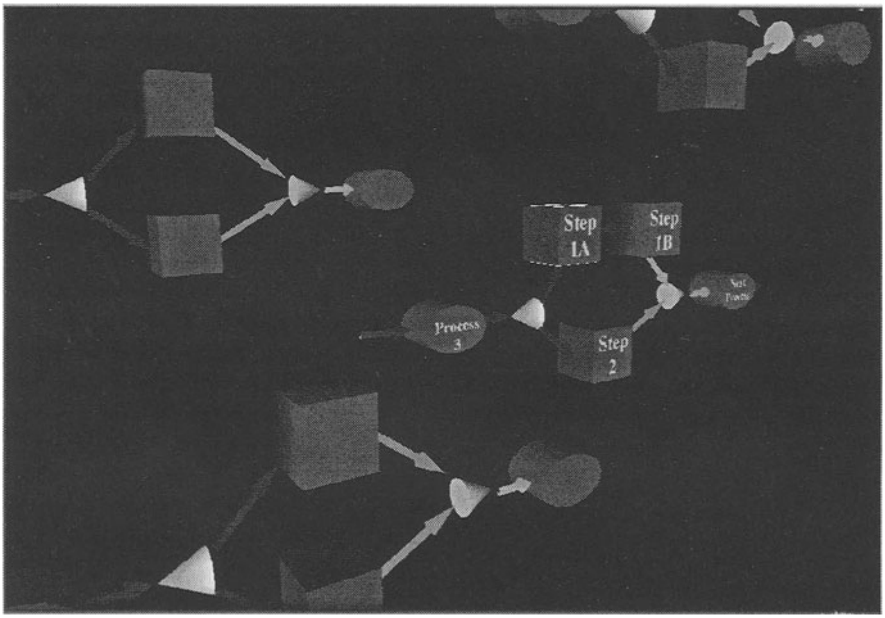


Fig. 1. Virtual Planning Environment.

The object oriented structure of the visualization and its underlying representation allows the users to 'drill down' into the nodes to further decompose the plan visualization, refine planning constraints, submit smaller goals to other planning systems, and to receive detailed logistical data. The virtual environment assimilates data from multiple distributed resources and systems into one consistent interface. This allows users to form a visual comparison to the plan and the real execution, thereby providing a more rapid awareness of plan deviations.

Intelligent Agents within the system automatically monitor the progress of the execution model versus the plan model and provide alerts when the deviation falls outside the set parameters. Some users will only be interested in specific types of information. The agents can be configured to search for information relevant to a user's interests or similarly for others users who have similar interests. Since the agents *live* in this virtual environment, they continuously monitor events and activities and can provide external notification services to users. Agents can further provide multiple levels of security to system resources. This allows for an expanded user base since certain privileges can be restricted to specific users.

The virtual environment provides a framework in which distributed users can collaborate and share resources, including a variety of multimedia types. This is an important consideration for large, complex tasks where responsibility lies with numerous, distributed individuals. Virtual user groups might develop where users with common interests could come to share, discuss, and archive their problems and solutions. This is a powerful tool for a user wishing to find information about a specialty area or obscure topic. Automated guided tours through the virtual environment could ease the task of orienting new users to the various systems and the multitude of available information.

3 Desk-top Virtual Reality

3D visualization is a key component of MAVE. When attempting to visually interface to and represent very complicated applications, such as planning systems, it is advantageous to utilize as many dimensions of communication as possible. This allows the end user to create, monitor, and control the application more effectively. The recent standardization of VRML (Virtual Reality Modeling Language) combined with the rapidly increasing power of the personal computer is allowing the technology of Modeling and Simulation to move from the domain of engineering applications, requiring expensive computer systems, to everyday desktop use [1].

VRML has many inherent strengths that make it an excellent choice for representing a distributed virtual environment. It runs within a web browser, supports JavaScript, provides a sophisticated API accessible from external JAVA applets/applications, and is hierarchical by design and therefore readily supports an object-oriented implementation. The current VRML standard already supports a wide array of multimedia including video, sound, and midi as well as multiple interface devices like the space mouse and 3D glove. These capabilities facilitate the 'fusion' of multiple, heterogeneous data sources into one consistent interface. The web-based architecture naturally lends itself to Internet-based collaboration methods.

4 Architecture

The cornerstone of MAVE is a novel 'end-to-end' object oriented design. Our research focuses on architectures at two different tiers. Figure 2 depicts an object-level architecture that is crucial to creating the powerful software needed to provide the functionality of advanced virtual environments. Figure 3 depicts the system-level architecture that is designed to provide the infrastructure necessary to support persistence, scalability, collaboration, and the distributed nature of virtual environments.

MAVE uses VRML as one element of the overall virtual environment. Although VRML is expressive, by itself it lacks the power and the structure to support advanced programmatic capabilities [6]. VRML models are typically constructed statically and accessed via a web server. The VRML scene is simply represented as an ASCII file containing the scene description. This scene is then interpreted locally, with VRML rendering software, and displayed to the user. Although this structure has produced the ability to develop '3D-web page' type content, it is insufficient to create a robust, dynamic, distributed virtual environment [6]. To accommodate these needs, we have developed a Virtual Environment Component (VEC) architecture (Figure 2) that maintains a physical granularity to mimic the logical decomposition of the respective elements of the virtual environment. We define a VEC as the lowest level element in a virtual environment that can stand alone as a useful entity. This is intentionally a flexible definition but one that, in practice, is easy to follow. Since each VEC can correspond to an instance in an object oriented database, it makes sense to maintain a level of granularity that has some significant cognitive value. Each VEC in the virtual environment has a corresponding VRML visualization. Bearing in mind that this representation can change dynamically from user interaction or a real-time data feed, it is clear that we must have some programmatic representation. The VEC architecture internalizes a programmatic representation of the VRML and then creates the visualization. In response to a stimulus that would change the VRML visualization, the programmatic structure is changed first and then the visualization is updated. This provides the necessary mechanism for maintaining persistence in the virtual environment and a coupling between Agent and User Interface objects. By creating this relationship between the VRML, Agent, and User Interface Objects, we

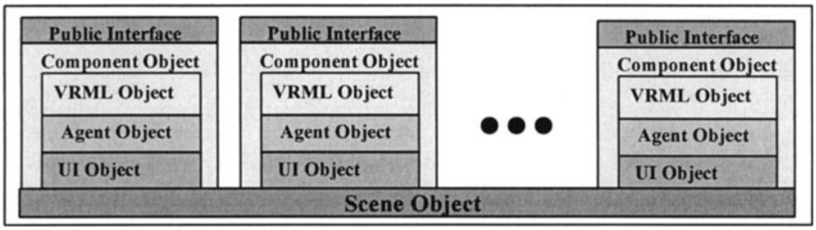


Fig. 2. Object Level Architecture.

are providing a mechanism for Intelligent Agents to obtain component level information that will allow them to provide better services to users.

The object oriented nature of this design will allow virtual environments to be flexible with both implementation and deployment issues. For example the Public Interface can be implemented with Java Beans or CORBA IDL technology and the distribution model can work with a CORBA ORB or a web server.

Figure 3 outlines the MAVE system-level architecture that is designed around an object-oriented database component. This component will augment the object-oriented structure of the virtual environment by providing a repository for persistent

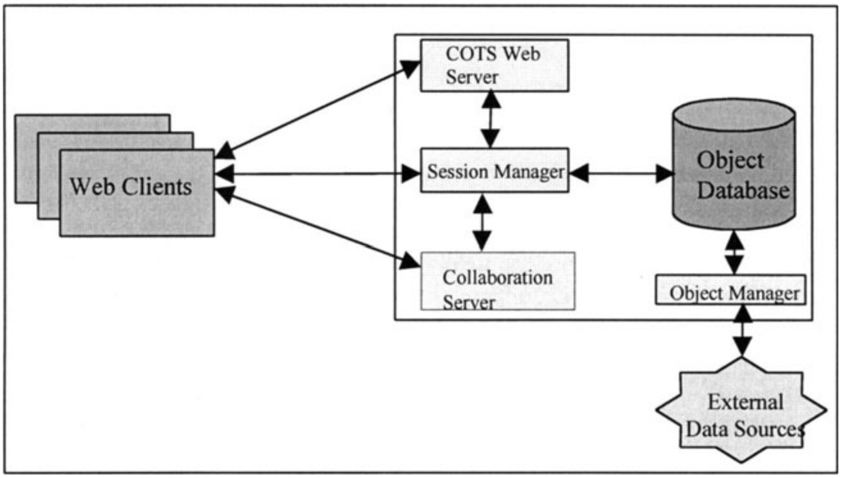


Fig. 3. Server Level Architecture.

VECs. An object-oriented database is a natural choice for the MAVE architecture due to the efficiency issues related to storing objects in a standard relational database. The COTS (Current Off The Shelf) Web Server provides a distribution mechanism for the VECs. For single server deployments, a web server would suffice, however for large, distributed implementations, the MAVE architecture will readily support CORBA products. The Session Manager has the basic responsibility of tracking users within the virtual environment and propagating the state of shared resources. The Collaboration Server is loosely defined here as it can function to provide services as basic as a text chat or as advanced as incorporating streaming video teleconferencing into the virtual environment. Lastly, the Object Manager is designed to interface with external data sources that can affect the internal state of the virtual environment or that may provide artifacts needed by the users. It will interface with the Object Database to update the state of the corresponding VECs. The advanced notification services of the database will then push the updates to the clients. This will allow for real-time state update of the virtual environment based on a real-time data feed.

5 Embedded Agents

One of our research goals is to reduce the level of difficulty faced by users who are required to access complex information systems. Even though a virtual environment does help to achieve this, adding Intelligent Agents into the structure will further improve the usability of systems.

Our use of Intelligent Agents focuses on two issues. The first is to develop agents that can provide intelligent assistance to users in the context of a virtual environment. Agents can provide navigation assistance to help users get where they want to be, assistance with locating information, and most importantly each VEC can employ its internal agent to assist users with accessing the particular services that it is designed to provide [2, 3]. Our other interest in agents comes from the desire to seamlessly integrate multiple VECs into one coherent interface. We are investigating methods by which the internal agent object of each VEC can publish interface specifications and provide services to other agents. This could help to abstract the user from some task decomposition and more fully fuse multiple resources and external systems.

The term 'Intelligent Agent' has become ubiquitous and has taken on a very vague definition. Our agent design includes some level of initial knowledge, the ability to perceive elements of the environment, and the subsequent ability to affect the environment [7], primarily through interaction with the user. The most important piece of the agent is the mapping between perception and the affect. This is accomplished by a reasoning or inference mechanism that is internal to the agent structure [7]. MAVE will support Intelligent Agents with virtually any type of reasoning method however we are focusing our research on the use of Bayesian Reasoning. Bayesian Reasoning is an appropriate choice for a virtual environment primarily because of the uncertainty associated with the environment and the probabilistic nature of the user interaction. Bayesian Reasoning, based on Bayes's Theorem [5], allows the agent to reach a rational decision even when there is not enough information to prove that any given action will [5, 7].

One of the primary reasons for developing the VEC architecture was to facilitate re-use of components. This produces a significant challenge to the agent embedded into a VEC in that the agent has no information about the context of the environment in which it will operate. The agent can provide a general interface that allows a virtual environment developer to provide some information, but a meticulous knowledge base is simply not practical. An agent employing Bayesian Reasoning can take a minimal amount of knowledge about its environment, combined with a learned probabilistic model of the user interaction, and still provide substantial assistance to the user. The ability to learn Bayesian causal relationships from data is a well established fact [4]. Our ongoing research is addressing the desire to combine the fragmented Bayesian networks, which are distributed across the VECs, into a more comprehensive representation of the environment and a more exhaustive probabilistic model of the user interaction.

6 Conclusions and Future Work

Our future work will focus on expanding the role of Intelligent Agents in MAVE. We will investigate the creation of a family of agents that can be easily incorporated into the VECs by virtual world developers. We will focus on methods for enhancing the knowledge acquisition skills of the agents embedded in the VECs and mechanisms by which the autonomy, interoperability, and user services of the VECs can be improved. We will investigate the applicability of machine learning to providing intelligent user assistance within virtual environments. Agents that observe user tasks can learn and abstract from the conclusions deduced. This could help to provide better assistance when users undertake new tasks that have yet to be observed by the agents.

In this paper, we have described an architecture with the ability to support a wide range of virtual environment applications. We are using VRML as a foundation for our virtual environment architecture, however, just like in the real world, deep meaning, information, and complex relationships cannot be represented by physical appearance alone. The VRML standard does not support expressive knowledge representations or the programmatic control to use them. MAVE is designed to utilize the power of VRML while addressing its shortfalls.

References

1. Braham, Robert and Comerford, Richard, *Sharing Virtual Worlds*, IEEE Spectrum, pp.18-25, March 1997.

2. Maes, P., *Intelligent Software*, Scientific American, Vol. 273, No.3, pp. 84-86, September 1995.

3. Maes, P., *Agents that Reduce Work and Information Overload*, Communications of the ACM, Vol. 37, No.7,pp. 31-40, 146, ACM Press, July 1994.

4. Mitchell, Tom, *Machine Learning*, New York, NY, The McGraw-Hill Companies, Inc, 1997.

5. Pearl, Judea, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, San Francisco, CA: Morgan Kaufmann Publishers, Inc, 1988.

6. Rockwell, Robert, *An Infrastructure for Social Software*, IEEE Spectrum, pp. 26-31, March 1997.

7. Russell, Stuart and Norvig, Peter, *Artificial Intelligence: A Modern Approach*, Englewood Cliffs, NJ: Prentice Hall Inc, 1995.

Agent-Based Simulation of Reactive, Pro-active and Social Animal Behaviour

Catholijn M. Jonker, Jan Treur

Vrije Universiteit Amsterdam
Department of Mathematics and Computer Science, Artificial Intelligence Group
De Boelelaan 1081a, 1081 HV Amsterdam, The Netherlands
URL: http://www.cs.vu.nl/~{jonker,treur} Email: {jonker,treur}@cs.vu.nl

Abstract. In this paper it is shown how animal behaviour can be simulated in an agent-based manner. Different models are shown for different types of behaviour, varying from purely reactive behaviour to pro-active and social behaviour. The compositional development method for multi-agent systems DESIRE and its software environment supports the conceptual and detailed design, and execution of these models. Experiments reported in the literature on animal behaviour have been simulated for a number of agent models.

1 Introduction

One of the most important aspects of agents (cf. [25]) is their behaviour. In the past, behaviour has been studied in different disciplines. In Cognitive Psychology the analysis of human behaviour is a major topic. In Biology, animal behaviour has been and is being studied extensively. Around 1900 a discussion took place about the manner in which observed animal behaviour can be interpreted in order to obtain an objective and testable description; for an overview, see [2], [23]. A risk of taking the intentional stance (e.g., [10]) as a perspective to explain behaviour, is that explanations are generated that make use of (a large number of) mental concepts that cannot be tested empirically. Therefore the *principle of parsimony* was introduced, stating that 'in no case may we interpret an action as the outcome of the exercise of a higher psychical faculty, if it can be interpreted as the outcome of the exercise of one which stands lower in the psychological scale'; see [18].

Building further on this perspective *behaviourism* was developed, e.g., [12], [21], [24]. In this approach animal behaviour is explained only in terms of a black box that for each pattern of *stimuli* (input of the black box) from the environment generates a *response* (output of the black box), that functionally depends on the input pattern of stimuli; i.e., if two patterns of stimuli are offered, then the same behaviour occurs if the two patterns of stimuli are equal. This view was also extended to human behaviour. Because of the underlying black box view, behaviourism discouraged reference to internal (mental) activities of organisms: any speculation about the internal functioning of the black box (i.e., the processes that might mediate between sensory inputs and behavioural outputs) was forbidden; cf. [23], p. 4.

In this paper, the compositional development method for multi-agent systems DESIRE (see [4]) is used to design, implement and experiment with agent-based simulation models for animal behaviour. In Section 3 a generic model of a *purely reactive agent* is introduced which is an adequate agent model to describe the (immediate) functional character of stimulus-response behaviour. The black box is represented by the agent component. The stimuli form the input (observation results), and the response is formed by the actions generated as output.

Viewed from a Software Engineering perspective, modelling behaviour by a functional relation between input and output provides a system that can be described as a (mathematical) *function* F : Input_states → Output_states of the set of possible input states to the set of possible output states. Such a system is transparent and predictable. For the same input always the same behaviour is repeated: its behaviour does not depend on earlier processes; for example, no information on previous experiences is stored in memory so that it can be remembered and affect behaviour. Well-known traditional programming methods are based on this paradigm; for example, program specification and refinement based on preconditions and postconditions as developed in, e.g., [11].

As opposed to behaviour defined by a purely functional dependency between input and output, an agent's behaviour often takes previous processes in which it was involved into account. These previous processes may have led to internal storage of information in a *memory* so that the same input pattern of stimuli can lead to different behaviour a next time it is encountered; the agent may be able to deliberate about it. Again viewed from a Software Engineering perspective, this makes that agents do not fit strictly in the paradigm based on a functional relation: to keep the functional relation, not only the actual input, but also the complete history of input should be taken into account, or the internal information in memory should be considered to be additional input.

In Section 4 a generic agent model is presented that can be used to model more complex behaviour. It includes not only components that represent the agent's memory (the agent's beliefs on the world and on other agents), but also components that represent the agent's goals and the agent's communication with other agents. This generic agent model has been used to obtain different agent models for different types of animal behaviour that go beyond purely reactive behaviour: *delayed response behaviour*, *deliberate pro-active behaviour*, and *social behaviour*. In Section 2 a problem description (a description of a pseudo-experiment) is presented; in Section 5 the behaviours of the different agent models introduced in Section 3 and 4 are compared for each of the situations defined in Section 2.

2 Problem Description

The deliberations put forward in the introduction can be illustrated by a very concrete example, taken from the discipline that studies animal behaviour; e.g., [23]. Animals, for example dogs, sometimes show a *delayed response*: they look for food in places where they have seen food before. This suggests that these animals might have some internal representation or memory of stimuli received earlier. More systematic experiments on this delayed response issue, for example those reported in [13] and [22], support this suggestion.

2.1 The Domain

The type of experiment reported in [22] is set up as follows (see Figure 1). Separated by a transparent screen (a window, at position p0), at each of two positions p1 and p2 a cup (upside down) and/or a piece of food can be placed. At some moment (with variable delay) the screen is raised, and the animal is free to go to any position. Consider the following three possible situations:

Situation 1 At both positions p1 and p2 an empty cup is placed.

Situation 2 At position p1 an empty cup is placed, and at position p2 a piece of food, which is (and remains) visible for the animal.

Situation 3 At position p1 an empty cup is placed and at position p2 a piece of food is placed, after which a cup is placed at the same position, covering the food. After the food disappears under the cup it cannot be sensed anymore by the animal.

In situation 1 the animal will not show a preference for either position p1 or p2; it may even go elsewhere or stay where it is. In situation 2 the animal will go to position p2, which can be explained as pure stimulus-response behaviour. In situation 3 the immediate stimuli are the same as in situation 1. Animals that react in a strictly functional stimulus-response manner will respond to this situation as in situation 1. Animals that show delayed response behaviour will go to p2, where food can be found.

In the literature, many reports can be found of observed delayed response behaviour in experiments of the type described above: [23], p. 4-5. The animal species used in these experiments vary from rats and dogs to macaques, chimpanzees and human infants. Therefore, it is assumed that animals of the type studied maintain internal (mental) representations on the basis of their sensor input, and that they make use of these representations (in addition to the actual sensory input that is used) to determine their behaviour. In a way it can be said that they may act as deliberate agents.

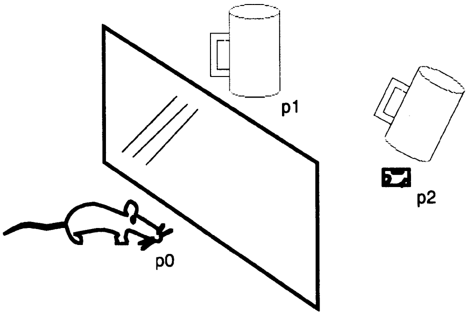


Fig. 1. Situation 3 of the experiment

2.2 The Requirements

In this paper four agent models A, B, C and D for the experiment are described. The following requirements on their behaviour express possible hypotheses that can be made about the behaviour of animals in the experiments:

A. An agent with purely reactive behaviour should behave the same for the two situations 1 and 3 described above: doing nothing, as if no food is present. Only in situation 2 should it go to the position of the food.

B. An agent with delayed response behaviour should behave the same in the situations 2 and 3: it should go to the position of the food. In situation 1 it should do nothing.

C. A deliberate pro-active agent's behaviour in the situations 1, 2 and 3 depends on whether the agent has a motivation or goal to do so. E.g., the agent may start acting in a pro-active manner (without any specific stimulus) in situation 1.

D. A social agent is able to take into account communication with other agents. If another animal is present that communicates that it wants to have the food (e.g., by growling), and the agent believes that this other agent is higher in the hierarchy, then the agent will not try to get the food.

3 An Agent Model for Purely Reactive Behaviour

An agent is *purely reactive* if it immediately responds to stimuli from its environment. Such agents are also called behaviour-based or situated agents; e.g., see [19]. These agents make their decisions based on a very limited amount of information, and simple situation-action rules. The stimuli can either consist of perceived changes in the external world or received communications from other agents. Changes in the external world are perceived by the agent by observation. The response behaviour of the agent affects its environment. Several archictures have been developed for reactive agents, see [1], [8], [16], [17]. In [19] an extensive overview of these architectures and the motivations behind them can be found.

3.1 Process Composition

For the design and implementation of the different models the compositional development method for multi-agent systems DESIRE has been used; see [4] for more details. A generic agent model for purely reactive behaviour developed earlier within the DESIRE environment (and applied in chemical process control) was reused. The (rather simple) agent system in this model consists of two components, one for the agent (of type A) and one for the external world with which it interacts (see Figure 2).

In the current domain, the observation information that plays a role describes that certain *objects* (cup1, cup2, food, screen, self) are at certain *positions* (i.e., p0, p1, p2). This is modelled by two sorts OBJECT and POSITION and a relation at position between these two sorts. Moreover, two types of *actions* can be distinguished: eat and goto some position. The latter type of actions is parameterized by positions; this can be modelled by a function goto from POSITION to ACTION. E.g., goto(p1) is the action to go to position p1. The action eat that is specified assumes that if the animal is at the position of the food, it can have the food: if a cup is covering the food, as part of the action eat the animal can throw the cup aside to get the food. Variables over a sort, e.g., POSITION, are denoted by a string, e.g., P, followed by : POSITION, i.e, P : POSITION is a variable over the sort POSITION. The unary relation to_be_performed is used to express the information that the agent has decided to *perform an action*; for example, to_be_perfomed(goto(p1)) expresses that the agent has decided to go to position p1. The relation observation_result is used to express the information that certain information has been acquired by *observation*; e.g., observation_result(at_position(food, p1), pos) expresses that the agent has observed that there is food at position p1, whereas the statement observation_result(at_position(food, p1), neg) expresses that the agent has observed that there is *no* food at position p1.

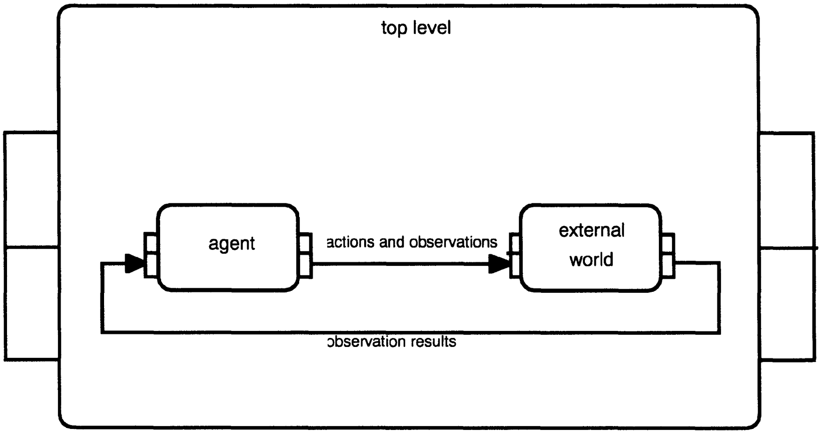


Fig. 2. A generic agent model for purely reactive behaviour

3.2 The Domain Knowledge

Assuming that food is offered at at most one position (for example, position p2), the stimulus-response behaviour of agent model A expresses that if the agent observes that there is food at any position and that no screen at position p0 separates the agent from this position, then it goes to this position. This knowledge has been modelled in the following form:

```
if    observation_result(at_position(food, P:POSITION), pos)
 and  observation_result(at_position(screen, p0), neg)
 and  observation_result(at_position(self, P:POSITION), neg) then to_be_performed(goto(P:POSITION))

if    observation_result(at_position(self, P:POSITION), pos)
 and  observation_result(at_position(food, P:POSITION), pos) then  to_be_performed(eat)
```

3.3 The Behaviour of the Purely Reactive Agent

The requirement imposed on agent A was that it shows the same behaviour for situations 1 and 3 in the problem description: do nothing. Moreover, in situation 2 the agent is required to go to the position of the food. The agent of type A indeed shows behaviour as expressed by the requirements.

4 Agent Models with More Complex Behaviour

To design an agent model that will show delayed response behaviour, the internal structure of the agent is made more complex. Within the agent a component maintenance of world information to maintain the observation results as beliefs (a memory) is distinguished from a component world interaction management that manages the interaction with the world. Moreover, if the agent has to generate its own goals in order to show pro-active behaviour, a component own process control is added, and if the agent has to show social behaviour, components are added to manage communication (agent interaction management) and to maintain beliefs on other agents (maintenance of agent information). The generic agent model depicted in Figure 3 (see also [7]) is composed of all of these components.

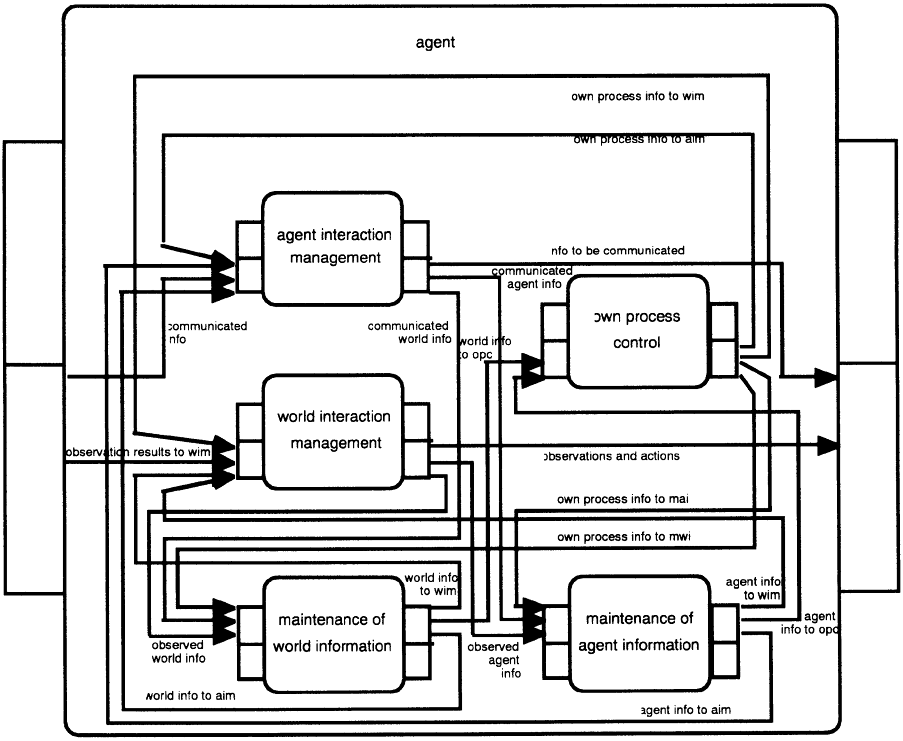


Fig. 3. A generic agent model for deliberate reactive, pro-active and social behaviour

4.1 An Agent Model with Delayed Response Behaviour

For an agent with *delayed response behaviour* (type B), the component maintenance of world information is used, in addition to the component world interaction management. The only task performed by the component maintenance of world information is storage of observation information. No further knowledge is used within this component.

The part of the knowledge of the component world interaction management that determines the actions is a variant of the knowledge used in agent model A. An additional part determines that the world information that was acquired by observation has *to be maintained*, expressed by the relation new_world_info.

```
 if      observation_result(I:INFO_ELEMENT, S:SIGN) then  new_world_info(I:INFO_ELEMENT, S:SIGN)

 if      belief(at_position(food, P:POSITION), pos)
  and  belief(at_position(screen, p0), neg)
  and  belief(at_position(self, P:POSITION), neg)    then        to_be_performed(goto(P:POSITION))

 if      belief(at_position(food, P:POSITION), pos)
  and  belief(at_position(self, P:POSITION), pos)    then        to_be_performed(eat)
```

An essential difference with the knowledge in agent model A is that in the knowledge above the relation observation result is replaced by the relation belief. Not only can information from direct observation be used, but also information retrieved from memory: all input information gets the status of belief, in contrast to observation. The behaviour of this agent model in comparison to the behaviours of the other models is discussed in Section 5.

4.2 An Agent Model with Pro-active Behaviour

The third agent model to be discussed is a model for a *pro-active agent* (type C). A pro-active agent does not simply respond to stimuli, neither immediately, nor delayed. In addition to the observation information, its so-called *motivational attitudes* (such as goals and plans) play an important role (e.g., see [25]). These motivational attitudes can be based on the agent's own character (for example, an agent's character may be that it always wants to eat, or that it is totally apathic), but also on specific aspects of the agent's own state, such as being hungry, or being depressed. To determine the motivational attitudes of the agent, the component own process control is used; additional knowledge structures are introduced for this new component. One action is added to the information type domain actions: the action explore position. In addition to the existing information types of agent model B, information types are required for knowledge on own process control; these information types express information on:

- the agent's beliefs
- aspects of the agent's own state, such as being hungry, or being depressed, and specific characteristics of the agent, such as always eager to eat, or totally apathic
- the agent's goals, such as be fed, just hang around, find food, or get food inside.

Information on the agent's *own state* can be expressed using the unary relation own_state; for example, the statement own_state(hungry) expresses that the agent is hungry. The agent's *own characteristics* can be expressed using the unary relation own_characteristic; e.g., the statement own_characteristic(totally_apathic) expresses the information that the agent is totally apathic. The *goal* that has been selected by the agent is expressed using the unary relation selected_goal. The knowledge to be used in the component own process control models:

- an agent that is always eager to eat, always selects the goal be fed
- any not apathic agent that is hungry or depressed selects the goal be fed
- a totally apathic agent always selects the goal just hang around
- an agent which has be fed as a goal, selects the goal get food inside if it has a belief that food is present at a specific position; in the other case it selects the goal find food

The knowledge used in own process control knowledge can formulated in a concise form as follows:

```
if      own_characteristic(always_eager_to_eat)   then     selected_goal(be_fed)

if      own_state(hungry)
  and not own_characteristic(totally_apathic)     then     selected_goal(be_fed)

if      own_state(depressed)
  and not own_characteristic(totally_apathic)     then     selected_goal(be_fed)

if      own_characteristic(totally_apathic)       then     selected_goal(just_hang_around)

if      selected_goal(be_fed)
  and belief(at_position(food, P:POSITION), pos)  then     selected_goal(get_food_inside)

if      selected_goal(be_fed)
  and not  belief(at_position(food, p1), pos)
  and not  belief(at_position(food, p2), pos)     then     selected_goal(find_food)
```

Depending on the type of agent modelled, some facts can be added to this knowledge base, for example in the agent of type C:

```
own_characteristic(always_eager_to_eat)
```

(alternatively, for example, own_state(hungry), **not** own_characteristic(totally_apathic) could be specified, or own_characteristic(totally_apathic))

Depending on the agent characteristics specified, the agent determines one or more goals. To actually show certain pro-active behaviour, also suitable knowledge has to be specified on which actions are to be performed for a given goal. This knowledge is used in the component world interaction management.

To determine actions related to the goal get food inside, two possible cases are considered:

- the agent believes that food is present at its own position; in this case it simply can start eating
- the agent believes that no food is present at its own position, but it believes that food is present at another position; in this case the agent can go to such a position (and if it arrives there it can start eating, according to the previous item)

This knowledge is expressed in a concise form as follows:

```
if      selected_goal(get_food_inside)
  and belief(at_position(food, P:POSITION), pos)
  and belief(at_position(self, P:POSITION), pos)    then      to_be_performed(eat)
```

```
if     selected_goal(get_food_inside)
  and belief(at_position(self, P1:POSITION), pos)
  and belief(at_position(food, P1:POSITION), neg)
  and belief(at_position(food, P2:POSITION), pos)
  and belief(at_position(screen, p0), neg)          then        to_be_performed(goto(P2:POSITION))
```

The goal get food inside assumes that the agent already knows at least one position where food is present. If this is not the case, the goal find food may be selected by the agent. To determine the actions for the goal find food, the following cases are considered:

- the agent does not know whether food is present at its own position; then the action explore position is selected (which determines whether food is present at the agent's own position)
- the agent believes that no food is present at its own position, and it does not know whether food is present at positions p1 and p2; in this case the action go to p1 is selected (and if it arrives there it can start exploring it, according to the previous item)
- the agent believes that no food is present at its own position and at position p1; it does not know whether food is present at p2; in this case the agent goto p2 is selected (and if it arrives there it can start exploring the position, according to the first item)

This knowledge is expressed in a concise form as follows:

```
if     selected_goal(find_food)
  and belief(at_position(self, P:POSITION), pos)
  and not belief(at_position(food, P:POSITION), pos)
  and not belief(at_position(food, P:POSITION), neg)   then   to_be_performed(explore_position)

if     selected_goal(find_food)
  and belief(at_position(self, P:POSITION), pos)
  and belief(at_position(food, P:POSITION), neg)
  and not belief(at_position(food, p1), neg)
  and not belief(at_position(food, p1), pos)
  and not belief(at_position(food, p2), neg)
  and not belief(at_position(food, p2), pos)          then        to_be_performed(goto(p1))

if     selected_goal(find_food)
  and belief(at_position(self, P:POSITION), pos)
  and belief(at_position(food, P:POSITION), neg)
  and belief(at_position(food, p1), neg)
  and not belief(at_position(food, p2), neg)
  and not belief(at_position(food, p2), pos)          then        to_be_performed(goto(p2))
```

4.3 An Agent Model with Social Behaviour

To obtain *social behaviour* (an agent model of type D), also the components agent interaction management and maintenance of agent information are used in the model. In the component agent interaction management knowledge is specified that identifies new communicated knowledge about other agents:

```
if communicated_by(I_want_food, pos, A:AGENT)  then        new_agent_info(wants_food(A:AGENT))
```

Here, the statement communicated_by(I_want_food, pos, A:AGENT) expresses that the information I_want_food *has been communicated* (positively) by the agent A:AGENT. This *new agent information* (expressed using the relation new_agent_info) is stored in the component maintenance of agent information: the knowledge used in maintenance of agent information specifies the hierarchy between different animals, and whether another animal that is present wants the food. For example, if an1 is an animal which is present and is *higher in the hierarchy*, then this can be specified within the knowledge base used in maintenance of agent information of the agent self as higher_than(an1, self). It is also possible to model this information in a dynamic form, as an outcome of earlier experiences (fights). If the other animal wants the food, within the component maintenance of world information of the agent self it is derived that the food is protected, using the knowledge

if wants_food(A:AGENT) **and** higher_than(A:AGENT, self) **then** food_protected

Within the knowledge elements used in the component world interaction management an additional condition **not** belief(food_protected, pos) is specified.

5 Overview of the Behaviour of the Different Agent Models

In the following table the differences in behaviour of the agent models are summarised. Here D2 denotes the agent model D in the presence of a higher competitor, whereas D1 denotes the same agent model without such a competitor.

	agent A	*agent B*	*agent C*	*agent D1*	*agent D2*
situation 1 (no food)	do nothing	do nothing	look for food	look for food	do nothing
situation 2 (visible food)	go to food	go to food	go to food	go to food	do nothing
situation 3 (invisible food)	do nothing	go to food	go to food	go to food	do nothing

The different variants of behaviour depicted in this table indeed satisfiy the requirements expressed in Section 2.

6 Conclusions

In this paper it is shown how different types of animal behaviour can be modelled and simulated at a conceptual level on the basis of the compositional multi-agent development method DESIRE (cf. [4]). Different (variants of) reusable compositional agent models were used to model the different required behaviours. The advantage of this approach is that the models are designed at a high conceptual level, in terms of the processes, information and knowledge that is needed, and abstracting from implementation details. Nevertheless they can be executed by the DESIRE software environment. Besides the simulation of animal behaviour discussed in this paper, a variety of other applications have been developed using DESIRE. Some recent multi-agent applications can be found in [3] (negotiation between agents), [5] (simulation of a society of agents), [6] (distributed work flow and agenda scheduling), and [15] (agents in brokering processes).

Simulation of animal behaviour is an interesting type of application for multi-agent systems. Both areas can benefit from a more extensive study of this type of application. The area of multi-agent systems can benefit from the more detailed analyses and distinctions that have been made for different types of animal behaviour (see the introduction). The study of animal behaviour can benefit from software tools for agent modelling at a conceptual level that support simulation. Moreover, formal techniques in the area of verification can be used to analyse and formalise behaviour properties of animals and their logical relations; e.g., [9], [14]. These formalisations can be a basis for the development of a theory of animal behaviour.

Acknowledgements

Parts of an earlier version of this paper were read by Frances Brazier. Based on her comments a number of improvements have been made in the text.

References

1. Agre, P.E., and D. Chapman, Pengi: an Implementation of a Theory of Activity. In: Proceedings of the sixth National Conference of the American Association for Artificial Intelligence (AAAI-87), Morgan Kaufmann, 1987, pp. 268-272.
2. Allen, C., and Bekoff, M., Species of Mind: the philosophy and biology of cognitive ethology. MIT Press, 1997
3. Brazier, F.M.T., F. Cornelissen, R. Gustavsson, C.M. Jonker, O. Lindeberg, B. Polak, J. Treur, Agents Negotiating for Load Balancing of Electricity Use. In: Proceedings of the Third International Conference on Multi-Agent Systems, ICMAS-98, IEEE Society Press, 1998
4. Brazier, F.M.T., Dunin-Keplicz, B., Jennings, N.R. and Treur, J., Formal specification of Multi-Agent Systems: a real-world case. In: V. Lesser (Ed.), Proceedings of the First International Conference on Multi-Agent Systems, ICMAS-95, MIT Press, Cambridge, MA, 1995, pp. 25-32. Extended version in: International Journal of Cooperative Information Systems, M. Huhns, M. Singh, (Eds.), special issue on Formal Methods in Cooperative Information Systems: Multi-Agent Systems, vol. 6, 1997, pp. 67-94.
5. Brazier, F.M.T., Eck, P.A.T. van, and Treur, J. Modelling a society of simple agents: from conceptual specification to experimentation. In: Conte, R., Hegselmann, R. and Terna, P. (eds.), Simulating Social Phenomena, Proceedings of the International Conference on Computer Simulation and Social Sciences, ICCS&SS'97, Lecture Notes in Economics and Mathematical Systems, vol. 456, Springer Verlag, 1997, pp. 103-107.
6. Brazier, F.M.T., C.M. Jonker, F.J.Jüngen, J. Treur, Distributed Scheduling to Support a Call Centre: a Co-operative Multi-Agent Approach. In: Proceedings of the Third International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAM-98, The Practical Application Company Ltd, 1998
7. Brazier, F.M.T., Jonker, C.M., Treur, J. , Formalisation of a cooperation model based on joint intentions. In: J.P. Müller, M.J. Wooldridge, N.R. Jennings (eds.), Intelligent Agents III (Proc. of the Third International Workshop on Agent Theories, Architectures and Languages, ATAL'96), Lecture Notes in AI, volume 1193, Springer Verlag, 1997, pp. 141-155.

8. Brooks, R.A., A robust layered control system for a mobile robot. In: IEEE Journal of Robotics and Automation, Vol. RA-2 (1), 1986, pp. 14-23.
9. Cornelissen, F., Jonker, C.M., Treur, J., Compositional verification of knowledge-based systems: a case study in diagnostic reasoning. In: E. Plaza, R. Benjamins (eds.), Knowledge Acquisition, Modelling and Management, Proceedings of the 10th EKAW'97, Lecture Notes in AI, vol. 1319, Springer Verlag, 1997, pp. 65-80.
10. Dennett, D.C., The Intentional Stance. MIT Press, Cambridge, 1987.
11. Dijkstra, E.W., A discipline of programming. Prentice Hall, 1976.
12. Gibson, J.J., The concept of the stimulus in psychology. In: American Psychology 15, 1960, pp. 694-703.
13. Hunter, W.S., The delayed reaction in animals. Behavioral Monographs, 2, 1912, pp. 1-85
14. Jonker, C.M., Treur, J., Compositional Verification of Multi-Agent Systems: a Formal Analysis of Pro-activeness and Reactiveness. In: W.P. de Roever, A. Pnueli et al. (eds.), Proceedings of the International Workshop on Compositionality, COMPOS'97, Springer Verlag, 1998.
15. Jonker, C.M., J. Treur, A Generic Architecture for Broker Agents. In: Proceedings of the Third International Conference on Practical Applications of Agents and Multi-Agent Systems, PAAM-98, The Practical Application Company Ltd, 1998
16. Kaelbling, L.P., An archictecture for intelligent reactive systems. In: Allen, J., and J. Hendler, and A. Tate (eds.), Readings in Planning, Morgan Kaufmann, 1990, pp. 713-728.
17. Maes, P. (ed.), Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back. MIT / Elsevier, 1990.
18. Morgan, C.L., An introduction to comparative psychology. London: Scott, 1894.
19. Müller, J. P., The design of intelligent agents: a layered approach. Lecture Notes in Artificial Intelligence, Vol. 1177, 1996.
20. Pavlov, I.P., Conditioned reflexes. London: Oxford, 1927.
21. Skinner, B.F., The generic nature of the concepts of stimulus and response. Journal of gen. Psychology 12, 1935, pp. 40-65.
22. Tinklepaugh, O.L., Multiple delayed reaction with chimpanzees and monkeys. Journal of Comparative Psychology, 13, 1932, pp. 207-243
23. Vauclair, J., Animal Cognition. Harvard Univerity Press, Cambridge, Massachusetts, 1996.
24. Watson, J.P., Psychology from the standpoint of a behaviourist. Philadelphia: Lippincott, 1919.
25. Wooldridge, M.J., and N.R. Jennings, Intelligent Agents: Theory and practice. In: Knowledge Engineering Review, 10(2), 1995, pp. 115-152.

A Fuzzy-neural Multiagent System for Optimisation of a Roll-mill Application*

Carlos A. Iglesias[†][**], José C. González[‡] and Juan R. Velasco[‡]

[†]Dep. de TSC e Ing. Telemática, E.T.S.I. Telecomunicación
Universidad de Valladolid, E–47011 Valladolid, Spain
{cif@tel.uva.es}
[‡]Dep. de Ingeniería de Sistemas Telemáticos, E.T.S.I. Telecomunicación
Universidad Politécnica de Madrid, E–28040 Madrid, Spain
{jcg,juanra}@gsi.dit.upm.es

Abstract. This article presents an industrial application of hybrid system: the development of a fuzzy-neural prototype for optimising a roll-mill. The prototype has been developed following an agent-oriented methodology called *MAS-CommonKADS*. This prototype has the original characteristic of being agent-oriented, i.e. each learning technique has been encapsulated into an agent. This multiagent architecture for hybridisation provides flexibility for testing different hybrid configurations. Moreover, the intelligence of the agents allows them to select the best possible hybrid configuration dynamically, for some applications whereas no best configuration for all the situations has been determined.

1 Introduction

This article presents a generic philosophy for developing hybrid systems by the usage of agent technology for encapsulating different symbolic and connectionist modules. In addition, we follow a methodological approach to hybrid systems, applying an extension of *CommonKADS* for multiagent systems development that takes into account hybridisation issues in the design model. This paper shows the benefits of this approach in an industrial application, the optimisation of the steel band production of a roll-mill.

The outline of this paper is as follows. In section 2 the rationale of using a multiagent system for hybridisation and the methodological approach that has been followed is discussed. Section 3 describes the proposed architecture for a multiagent fuzzy-neural model. Section 4 shows how this generic architecture has been applied to an industrial application and the experimental results. Finally, some conclusions are drawn in section 5.

[*] This research is funded in part by the Commission of the European Community under the ESPRIT Basic Research Project *MIX: Modular Integration of Connectionist and Symbolic Processing in Knowledge Based Systems*, ESPRIT-9119, and by the Spanish Government under the CICYT projects TIC91-0107 and TIC94-0139.

[**] This research was partly done while the first author was visiting the Dep. Ingeniería de Sistemas Telemáticos (Universidad Politécnica de Madrid).

2 An agent-oriented methodological approach

The approach to hybrid systems development in the MIX project [GVI95] has consisted on encapsulating the learning components into agents. This provides a *component-based* approach to hybrid system design, since the interfaces between the components are standardised. The designer of the hybrid system can select the components (agents) to combine. The intelligence of the agents allow that the agents themselves can select their configuration.

The development of the fuzzy-neural prototype has followed the agent-oriented methodology *MAS-CommonKADS* [IGGV97]. This methodology allows the analysis and design of application from an agent-oriented perspective.

This methodology extends the knowledge engineering methodology *CommonKADS* [SWV94] with techniques from object-oriented and protocol engineering methodologies. The methodology consists of the development of seven models: *Agent Model*, that describes the characteristics of each agent; *Task Model*, that describes the tasks that the agents carry out; *Expertise Model*, that describes the knowledge needed by the agents to achieve their goals; *Organisation Model*, that describes the structural relationships between agents (software agents and/or human agents); *Coordination Model*, that describes the dynamic relationships between software agents; *Communication Model*, that describes the dynamic relationships between human agents and their respective personal assistant software agents; and *Design Model*, that refines the previous models and determines the most suitable agent architecture for each agent, and the requirements of the agent network.

The analysis phase [IGVE96,IGV96] has consisted of the development of a generic task model for fuzzy-neural hybridisation, the identification and description of the agents for carrying out these tasks in the *agent model*, and the description of the interactions and services between the agents in the *coordination model.* The tasks that require knowledge have been described in an *expertise model.*

The design of these knowledge tasks has taken into account the possibility of integrating hybrid techniques. The problem of integrating machine learning issues in *CommonKADS* has been tackled in [VdVA92,HLA95]. The approach has consisted of identifying the different learning methods that can be associated with an inference source [Hil95] and identifying learning goals [VdVA92] (e.g. tuning fuzzy sets) and defining learning tasks to achieve these goals and assign these tasks to learning agents. When there is several ways of performing a learning task, and there is no best known method, we have identified several strategies:

- An implementation of all the possibilities (or the most promising alternatives) into one intelligent agent, and the intelligent agent selects at run-time the most suitable method for performing the task. We have to define explicit criteria for selecting a method. In other case, these criteria must be learnt at run-time.

- Definition of one agent per method, and definition of an additional agent which selects the best result. That is, all the methods are executed in parallel, and the best one is selected in each execution (*stand-alone model*).
- A refinement of the previous method, called the *intelligent stand-alone model.* The intelligent stand-alone model is a compound of an agent *manager* and a set of learning agents which are included into a public group. The *manager* selects the best system or knowledge base that has been learnt. It can also select the different regions on which the learning agents should work. Depending on the results obtained by the different learning agents, the *manager* can learn which agents work better under certain circumstances. So, this *manager* performs meta-processing for determining the regions (macro-states) on which the learning agents will work.

The main benefits of the *intelligent stand-alone model* are that the *manager* is able to perform a meta-processing according to predefined beliefs (e.g. knowledge about different regions of the problem and knowledge) and dynamically learn and update its beliefs. The manager can also try different strategies to achieve its goals, as getting a new definition of the regions, not working anymore with regions, working with some knowledge base, not working anymore with some learning agent, etc.). This approach changes the traditional approach of manual testing, providing high flexibility for defining experiments.

In addition, the *learning agents* are able of evaluating the results they obtain. An interesting and useful feature is that the *learning agents* are able of changing their own parameters to achieve their goal (getting a better performance). For instance, our C4.5 agent (section 3) is able to modify the confidence level parameter with the strategy *generate-and-test.*

3 Architecture of the generic multiagent fuzzy-neural model

The main characteristics of this model are:

- Two learning tasks are identified: improving fuzzy sets and learning fuzzy rules. For each learning task, an *intelligent stand-alone model* is used, consisting of a manager and a set of learning agents.
- The combination of the fuzzy sets and the fuzzy rules is carried out by another *intelligent stand-alone model.* The manager of this configuration is called the *FZ_Engine* agent and the learning agents are the managers of the previously defined learning tasks (i.e. improving fuzzy sets and learning fuzzy rules) as shown in figure 1.
- The agent *FZ_Engine* carried out a partition of the complete problem space into a set of regions. The identification of regions can be carried out by neural nets and by fuzzy clustering algorithms.
- The agent *FS_Manager* is the manager of the fuzzy set improvement task. These fuzzy sets can be extracted by neural net agents (e.g. *NN_Get_FS_n* and fuzzy clustering agents (e.g. *FZ_Get_FS_n*) , in on-line or off-line mode.

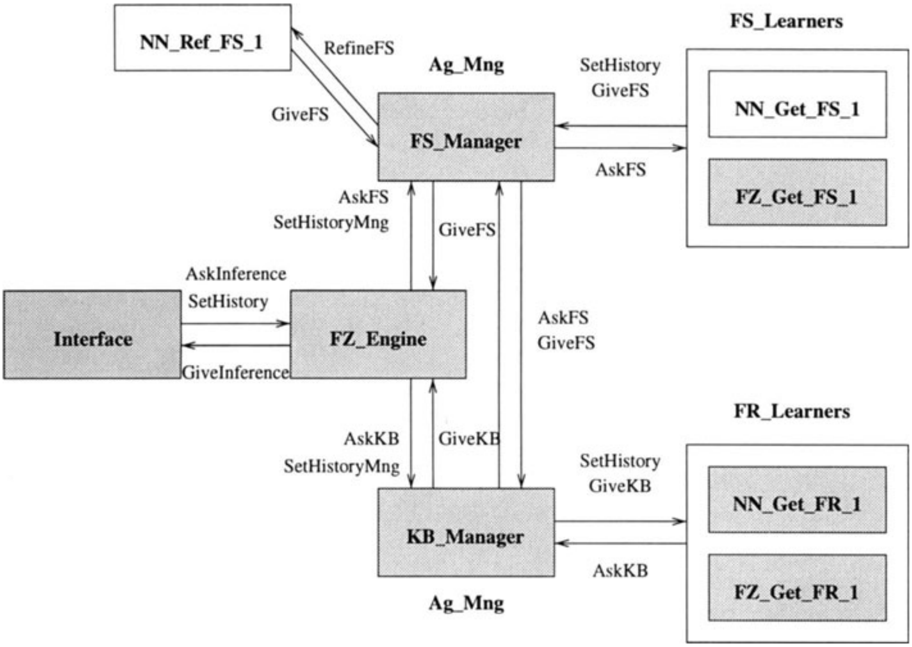


Fig. 1. Event flow diagram of the generic model. Shadowed boxes show the implemented prototype.

- The agent *KB_Manager* is the manager of the fuzzy rule extraction task. These rules are acquired inductively by neural nets agents (e.g. *NN_Get_FR_n*) or symbolic induction (e.g. *FZ_Get_FR_n*) in each region of the problem space, in on-line or off-line mode.

The hybrid systems that deal with fuzzy sets and fuzzy knowledge bases are, in their turn, built from an association/cooperative model: the so-called *intelligent stand-alone model.* The agents defined for this fuzzy-neural model will be reused for other multi-hybrid models (fuzzy logic, case based reasoning and neural networks) [ALLM96] in the different applications of the MIX project.

4 The roll-mill application

The Roll-mill application [Ale95,PA96] has been selected to test the generic hybrid model. In such process, the output of the intelligent system is the suggested rolling force to optimise the steel band production. Twelve parameters are used as the input of the force predictor: three discrete and nine continuous.

A data file with 11054 vectors (each vector with twelve inputs and the best output for them) has been used to train the system. 8866 vectors (80.2% data) constituted the train set, and the rest (2188), the test set. The value to optimise is the relative error as described in [Ale95].

Three experiments have been carried out: a pure connectionist system with unclassified data (section 4.1), a pure connectionist system with clusters (section 4.2) and a multiagent fuzzy-neural prototype (section 4.3).

4.1 Pure Connectionist System with Unclassified Data

A simple neural net was trained with the complete training set. For this purpose, a fully connected feed-forward Multi-Layer Perceptron with four layers (12*12*6*1) was used. Learning was achieved by using the back propagation algorithm (1000 epochs and a learning rate of 0.2). The average error resulting from this experiment was 12.1%.

4.2 Pure Connectionist System With Clusters

Three of the 12 input values, N_PASS, LUBRICATION and ROLL_FINISH, were discrete, the first two having 3, and the last one having 21 different values. So, it was probable that these values had a significant meaning for the process. To find out about this, the learning and test data sets were divided into six classes with different combinations of the two first discrete variables (there were no samples with N_PASS=0.5). Then, a net with the same architecture was trained for each of these sets and the nets were tested with the test sets of the corresponding class. The average error obtained was 12.3% (slightly higher).

By examining the data graphically, it could be seen that two of the input variables, WIDTH and THICKNESS, showed a higher correlation with the output variable than the rest. So, these two were selected as the basis for clusterisation. By using the ISODATA algorithm, the bi-dimensional problem space was divided into 7 classes as shown in table 1, together with the sizes of the corresponding data sets.

CLASS	THICKNESS	WIDTH	SIZE	
			Train	test
1	[0.00, 0.50]	[0.00, 0.30]	1310	334
2	[0.00, 0.30]	[0.30, 0.55]	1971	458
3	[0.00, 0.50]	[0.55, 0.75]	2044	522
4	[0.00, 0.50]	[0.75, 1.00]	1214	276
5	[0.50, 1.00]	[0.50, 1.00]	971	253
6	[0.50, 1.00]	[0.00, 0.50]	748	190
7	[0.30, 0.50]	[0.30, 0.55]	608	155

Table 1. Data clusterization

A net with the same architecture as before was trained for each of these classes and tested with the test data of the class. Results are shown in table 2

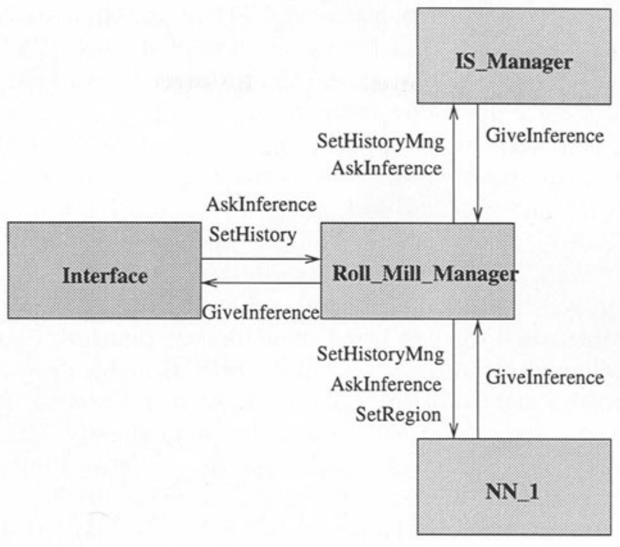


Fig. 2. Event flow diagram of the roll-mill model. The *IS_agent* is developed in fig. 1.

(column 2). The table shows that only the results for class 6 were better than with the initial network (column 3), but the average for these classes was worst.

The Stuttgart Neural Nets Simulator (SNNS)[1] was used for the implementation of pure connectionist systems.

4.3 Multiagent fuzzy-neural prototype

The generic architecture describe in section 3 has been instantiated for the roll-mill application. The prototype only considers some of the agents of the generic architecture, marked with shadowed boxes (figure 1). This model has been extended with another intelligent stand-alone configuration shown in figure 2. This configuration consists of an agent manager, a connectionist agent with clusters (*NN_1*), described in section 4.2, and an *intelligent stand-alone model IS_Manager* despicted in figure 1 and described below. The *MAST* multiagent platform[2] has been used for the implementation of the multiagent system.

The *FZ_Get_FS_1* agent uses the ISODATA fuzzy clustering algorithm to define trapezoidal fuzzy sets for every continuous variable of the system. This iterative algorithm classifies data according to their distance to a characteristic point of the cluster called centroid, based on the minimisation of the cluster volume and maximisation of its density. Centroids has been selected as the most

[1] The *SNNS* Stuttgart Neural Nets Simulator has been developed by the University of Stuttgart, at `http://vasarely.informatik.uni-stuttgart.de/snns/snns.html`

[2] The *MAST* multiagent platform has been developed by the Technical University of Madrid and it is available at `http://www.gsi.dit.upm.es/~mast`

representative peaks of the data histogram. The algorithm is based on the minimisation of the sets volume and maximisation of density [VV94,GG89]. For instance, the fuzzy sets obtained by the algorithm for the variables width of the band and rolling force are shown in figure 3.

The *FZ_engine* agent uses the fuzzy engine FuzzyCLIPS 6.02A[3]with standard max-min as compositional rule of inference and COG (Centre of gravity algorithm) algorithm for defuzzification.

The *FS_Get_FS_1* agent extracts the fuzzy rule using the C4.5 algorithm with pre-processed inputs and outputs as described in [IGV96]. For rule extraction C4.5 algorithm has been used as follows . First, historic numeric data file has to be transformed in a pseudo-fuzzy data file, by changing every number in the file by the label of the fuzzy set with bigger membership value. This data file feeds the C4.5 algorithm to create an Induction Decision Tree and a set of rules. C4.5 algorithm, that has been designed to classify data, can be used to create prediction systems (as in this case) if predicted values are discrete. In this case, predicted values are the fuzzy sets of the predicted variable. We have generated two knowledge bases of fuzzy rules. Each knowledge base has a different confidence factor. This factor sets the pruning level in the decision tree (intermediate step to create rules) and rules generation. The used factors are 25 (default) and 50.

The *NN_Get_FR_1* agent uses a Kohonen self-organising map, following the method for rule extraction described in [Kos92,Hun93], though the results have been worse in this application than the obtained with the *FZ_Get_FR_1* agent.

The table 2 shows the results obtained for the neural nets and fuzzy knowledge bases generated. The fuzzy KBs have been tested using the specific (S) generated fuzzy KB for every cluster and the general (G) fuzzy KB generated for the whole data. In addition, the results are shown for the values 25 and 50 of the confidence factor of C4.5.The last column remarks the best value of the error rate for each cluster, with the global mean error for these cases.

Data	Neural Nets		Fuzzy KBs				**Best**
	Spec.	Gen.	S-25%	S-50%	G-25%	G-50%	
C1	14.1	13.8	9.48	10.19	8.60	**8.43**	8.43
C2	14.6	14.1	13.43	5.90	5.82	**5.12**	5.12
C3	12.1	11.1	**5.93**	8.41	8.25	7.03	5.93
C4	14.7	13.2	7.31	12.67	9.79	**6.50**	6.50
C5	11.1	10.8	7.49	5.47	5.86	**5.81**	5.81
C6	**8.7**	9.6	9.93	11.36	13.32	10.35	8.70
C7	11.1	10.0	13.64	13.10	11.48	**7.03**	7.03
Global	12.77	12.16	9.29	8.94	8.38	6.92	**6.49**

Table 2. Average error over clusterised and global data

[3] FuzzyCLIPS has been developed by the Knowledge Systems Laboratory, National Research Council, Canada, available at `http://ai.iit.nrc.ca/fuzzy/fuzzy.html`

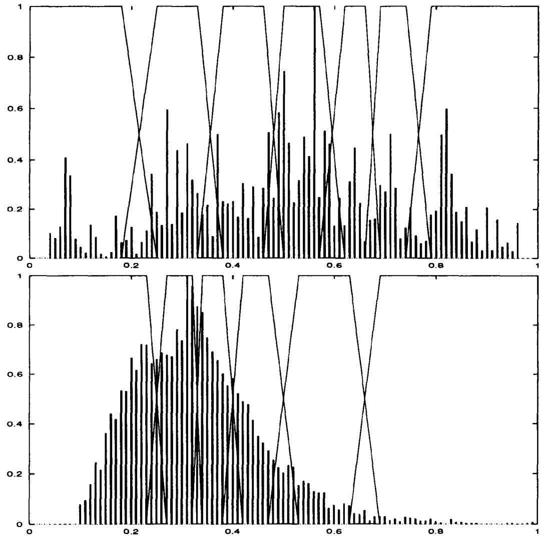


Fig. 3. *Width of the band* and *Rolling force* fuzzy set distributions

5 Conclusions

This article has shown a distributed approach to hybridisation that takes advantage of the intelligence of the components for determining the hybrid configuration at run-time. The application of a methodology contributes to the reuse these configuration and the developed agents.

The main results obtained in this work from a conceptual point of view are:

- The adoption of a methodology for developing hybrid systems for an agent-oriented perspective.
- The definition of a reusable configuraton called the *intelligent stand-alone model* for building hybrid systems.
- The design and implementation of a hybrid fuzzy-neural model which is a first attempt towards an association model. This model has the additional advantage of being open, and it can be easily combined with other paradigms.
- The design and implementation of a reusable agent class Selector for intelligent stand-alone models.
- A reusable implementation of services for learning agents.
- The implementation of reusable fuzzy and neural agents. These agents have been reused in a fuzzy logic, case based reasoning and neural networks hybrid model [ALLM96].

From the analysis of performance of this experience, these results can be remarked :

- We have obtained better results than with the pure neural approach. The error rate has been reduced to 50%.
- Our premise of getting better results dividing the data space in regions seems to be promising. More extensive work has to be done, however, to determine a better splitting of the problem space into regions by using both symbolic and connectionist methods.
- More extensive work is needed in the neural part for extracting fuzzy sets and rules.
- Fusion models need to be validated and integrated in our model.

Acknowledgements

We would like to thank all the members and collaborators of the MIX project. They have contributed to a very fruitful and friendly atmosphere for researching.

References

[Ale95] Frederic Alexandre. Report on definition and specification of industrial applications: Steel rollmill application. Technical Report MIX/WP3/INRIA/1.0, INRIA Lorraine, March 1995.

[ALLM96] F. Alexandre, A. Labbi, Y. Lallement, and M. Malek. Evaluation report on the case-based reasoning / neural networks hybrid architecture. Technical Report Deliverable D8, IMAG - INRIA, 1996.

[GG89] I. Gath and A. B. Geva. Unsupervised optimal fuzzy clustering. *Trans. on Pattern Analisys and Machine Intelligence*, PAMI-11(7):773–781, July 1989.

[GVI95] José C. González, Juan R. Velasco, and Carlos A. Iglesias. A distributed platform for symbolic–connectionist interoperation. In Ron Sun and Frederic Alexandre, editors, *Proceedings of the IJCAI'95 Workshop on Connectionist-Symbolic Integration: From Unified to Hybrid Approaches*, pages 94–99, Montreal, Canada, August 1995. International Joint Conference on Artificial Intelligence. (An extended version of this paper has appeared in *Connectionist Symbolic Integration*, Lawrence Erlbaum Associates, 1996.).

[Hil95] Melanie Hilario. KADS: A framework for neurosymbolic integration? Technical Report UNIGE-AI-95-04, Computer Science Department. University of Geneva, 1995.

[HLA95] M. Hilario, Y. Lallement, and F. Alexandre. Neurosymbolic Integration: Unified versus Hybrid Approaches. In M. Verleysen, editor, *Proceedings European Symposium on Artificial Neural Networks*, Brussels (Belgium), 1995.

[Hun93] C. Hung. Building a neuro-fuzzy learning control system. *AI Expert*, 8(11):40–49, 1993.

[IGGV97] Carlos A. Iglesias, Mercedes Garijo, José C. González, and Juan R. Velasco. Analysis and design of multiagent systems using MAS-CommonKADS. In AAAI'97 Workshop on Agent Theories, Architectures and Languages, Providence, RI, July 1997. ATAL. (An extended version of this paper will be published in *INTELLIGENT AGENTS IV: Agent Theories, Architectures, and Languages*, Springer Verlag, 1998,.

[IGV96] Carlos A. Iglesias, José C. González, and Juan R. Velasco. Prototype of a fuzzy-neural hybrid model. evaluation report. Technical Report MIX/WP2/UPM/3.0, Dep. Ingeniería de Sistemas Telemáticos, E.T.S.I. Telecomunicación, Universidad Politécnica de Madrid, January 1996.

[IGVE96] Carlos A. Iglesias, José C. González, Juan R. Velasco, and Klaus Eder. Prototype of a fuzzy-neural hybrid model. Implementation report. Technical Report MIX/WP2/UPM/2.0, Dep. Ingeniería de Sistemas Telemáticos, E.T.S.I. Telecomunicación, Universidad Politécnica de Madrid, January 1996.

[Kos92] Bart Kosko. *Neural Networks and Fuzzy Systems. A Dynamical Systems Approach to Machine Intelligence.* Prentice-Hall, Englewood Cliffs, 1992.

[NK95] Detlef Nauck and Rudolf Kruse. NEFCLASS - a neuro fuzzy approach for the classification of data. In *Symposium on Applied Computing during 1995 ACM Computing Week*, Nashville, March 1995.

[PA96] Nicolas Pican and Frédéric Alexandre. Artificial neural networks for the presetting of a steel temper mill. *IEEE Expert*, pages 22–27, February 1996.

[SWV94] A. Th. Schreiber, B. J. Wielinga, and J. M. Akkermans W. Van de Velde. CommonKADS: A comprehensive methodology for KBS development. Deliverable DM1.2a KADS-II/M1/RR/UvA/70/1.1, University of Amsterdam, Netherlands Energy Research Foundation ECN and Free University of Brussels, 1994.

[VdVA92] Walter Van de Velde and Agar Aamodt. Machine learning issues in CommonKADS. Technical report ESPRIT Project 5248 KADS-II/TII.4.3/TR/VUBS/002/3.0, Free University Brussels, January 1992.

[VV94] Juan R. Velasco and F. Enrique Ventero. Some applications of fuzzy clustering to fuzzy control systems. In *Proceedings of the Joint Conference on Information Sciences*, pages 363–366, Pinehurst, NC, USA, November 1994.

High-Level Communication Protocol in a Distributed Multiagent System

Wided LEJOUAD-CHAARI & Fériel MOURIA-BEJI

Laboratoire d'Intelligence Artificielle - LIA
ENSI
Rue des Entrepreneurs
2035 LA CHARGUIA II - Tunis
TUNISIA

Abstract. *In previous multiagent system studies, the agents interact locally (on a single processor). In this paper, we consider the organization of these agents in a physical distributed universe, we outline the management of their interactions, and we propose the way to detect and resolve conflicts. We emphasize particularly the communication aspects considered as the basis of the cooperation, coordination, and negotiation mechanisms. Agents must deal with their respective tasks solely and leave any communiation management to a specialized agent which will be responsible for any problem that may arise. Since the specialized agents need to have a global view of the system state, we define a communication model which takes into account all possible conflicting cases and the established communication models and modes.*

1 Introduction

A Distributed Multiagent System (DMS) consists of multiple processes, called intelligent agents, which are physically separated. These agents coordinate their actions, cooperate, and negotiate some decisions to solve a given problem. Previous multiagent system experiments [7], [6], [11], [12] use traditional communication protocols, called low-level protocols, which offer a reliable and efficient transmission of bit streams between agents, but they do not consider the semantics of the information being passed. The aim of this paper is to provide a distributed multiagent system with a high-level communication protocol which assigns interpretation to the bit streams, and offers a model to present *what* the agents should say to each other, rather than *how* they say it. This communication semantics has to be expressed independently of the agent functionalities, in order to ensure the reuse of both components and to make the model growing up without changing the intrinsic agent behavior.

The costs of communications are expensive and so, reduce the system performance. Thus, communication cannot be neglected in the design of a distributed problem-solving. Among the major issues involved in the construction of a DMS, is the specification of communication policies so that agents can interact easily. In this paper, we present a new architecture in which agents include a communication layer that allows them to perform a low-level message traffic. The high-level communication is supported by a particular agent called Communication Manager (CM). Each group of interacting agents will be associated with a CM that will provide a metaknowledge about what kind of information do agents exchange, are there conflicts on shared data, which communication modes have been used, are there deadlocks, how to resolve them, ...

In this paper, we start by defining distributed multiagent systems, then we describe the communication model expressed in the communication manager agent, we also define the global architecture made of multiple CMs, each one associated with a collection of remote intelligent agents. Finally we give a conclusion to define the platform used to generate agents, to implement the model and to provide a flexible and dynamic architecture integrating the distributed system concepts.

2 Distributed Multiagent System

We define a distributed multiagent system as a collection of autonomous agents which are logically and physically separate. Each agent includes a Source Knowledge (SK), a Data Base (DB) and a reasoning which makes it able to solve a specific sub-problem. In this section, we give the key characteristics of a DMS [5].

- *Knowledge sharing*: agents working together may need access to each other's knowledge and can share the same set of data. For effective sharing, each agent must be managed by a program that offers a communication interface enabling its knowledge source to be accessed, manipulated and updated reliably and consistently.

- *Openness*: it determines whether the system can be extended. This extension can concern communication protocols, resource-sharing services and system evolution by adding new intelligent agents.

- *Concurrency*: when several agents access the same resource concurrently, a particular process must synchronize their actions to ensure that they do not conflict. The synchronization must be carefully planned to ensure that the benefits of concurrency are not lost.

- *Transparency*: the DMS is perceived as a whole rather than as a collection of independent agents. It enables local and remote information objects to be accessed using identical operations, and allows the transfer of information objects within a system without affecting the agent behaviors [2].

- *Consistency*: a DMS is faced with a lot of difficulties, such as, conflict resolution of concurrent access to shared data, deadlocks, bottelnecks, and so forth. Mechanisms and methods dealing with these problems have to be considered in order to provide a coherent solution for the whole problem.

- *Fault tolerance*: computer systems sometimes fail. When faults occur in the hardware or the software, a DMS may produce incorrect results or it may stop before it has completed the intended task. Methods for detecting and recovering from these faults must be provided.

In Distributed Systems, these characteristics are assumed by communication protocols. In this paper, we try to focus on the independence of these functionalities from the internal behavior of the agent. The Communication Manager is the only agent able to resolve problems, due to the distribution, occuring during the system runnig.

3 Communication Model

Among the major issues involved in the construction of a distributed artificial intelligence system, is the specification of communication policies so that the processing agents can interact and cooperate when no one of them has sufficient information to solve the entire problem. This sharing of information is necessary to allow the system as a whole to produce a consistent answer.

In general, communications between a pair of agents can be viewed as *message-passing* actions, each one involves the transmission by the sending agent of a set of data values (a message) through a specified communication mechanism (a channel or port) and the acceptance by the receiving agent of the message. There are three types of messages communicated among agents: a *query-type*, an *answer-type* and an *information-type*.

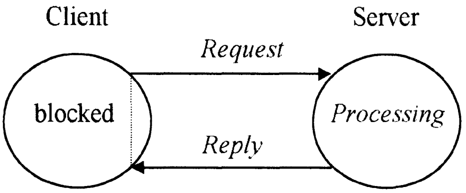


Fig.1. Client-server communication

These messages can be exchanged in a *client-server* communication model, a *group multicast* communication model, or both. The first model is well adapted to a direct transmission between pairs of agents (see figure 1), while the second is largely used in a broadcast transmission between groups of cooperating agents (see figure 2). In the design of our distributed multiagent system (see figure 3), we choose the last model which groups the most frequently used means of communications. The communication mechanism may be *synchronous* (blocking), meaning that the sender waits after transmitting a message until the receiver has performed a receive operation or it may be *asynchronous* (non-blocking), meaning that the message is placed in a queue of messages waiting for the receiver to accept them and the sending agent can proceed immediately.

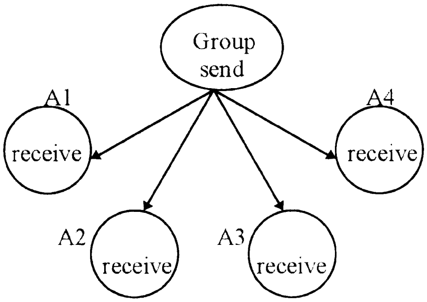


Fig. 2. Multicasting to an agent group

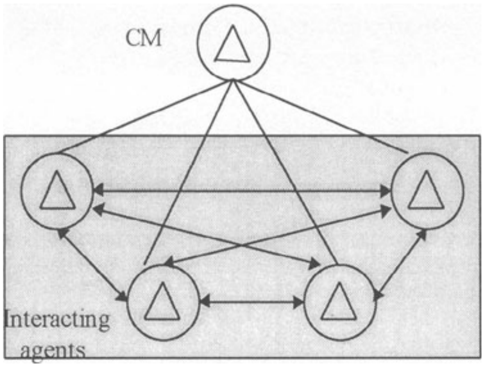


Fig. 3. Mixed communication model

The model presented here considers the case where agents act (negotiate) to resolve inconsistent views and to reach agreement on how they should work together in order to cooperate effectively.

4 Metaknowledge in the Communication Manager

In this section, we define the metaknowledge specification in the CM agent. This one must describe the transmission mode between agents, the type of message exchanged, their actions and reactions, and it must also detect and manage failure cases and deadlocks that may occur. The main difficulty in the CM specification resides in the definition of the basic properties the model might include to provide a high-level communication in a DMS.

```
-- CM between Agents Ai
CM (A1, ..., Ai, ..., An)
{
    Message → From AgentId;
              To   AgentId;
              Type MesgT;
               Mode TransM;
               Model ComM;

    AgentId → IPaddress | SymbolicId;
    MesgT   → Query | Answer | Information;
    TransM  → Synchronous | Asynchronous;
    ComM    → Client-server | Multicast;

    Query → QueryId;
            -- Receive Agent : R-Agent
            -- Check Status : C-Status
            if  R-Agent exists
            then C-Status (R-Agent)
            else CancelMesg (QueryId)
            endif;

    C-Status (AgentId) → if  not-free
                         then Wait until free
                          -- Check Sharing : C-Sharing
                          else C-Sharing
                          endif;
    C-Sharing → -- Transactions Tj in read-write
                -- locking or write-write locking on the
                -- same data item
                if Ti & ... & Tj [LRW | LWW ] DItem
                 then Conflict
                endif;
                Exec (Mesg);
```

```
Conflict →  call Locking-Algorithm;
            if Deadlock
             then call Deadlock-Algorithm
             endif;
Ti → Transaction from Agent_i ;
Transaction → {RO,WO}+;
RO → Read Operation;
WO → Write Operation;
LRW → Read & Write Locks;
LWW → Write & Write Locks;
DItem → Shared Data Item;

Answer →  To QueryId;
          -- Check Time : C-Time
           C-Time;
           -- Sending Agent :  S-Agent
           if  S-Agent (QueryId) exists
          then C-Status (S-Agent)
          else CancelMesg (QueryId)
           endif;

C-Time → if time(answer) > Timeout
         then CancelMesg (QueryId)
         endif;

Synchronous → Send (Mesg);
              Wait (Mesg);
              Receive (Mesg);
              S-Actions;

Asynchronous → Send (Mesg);
               Continue;
               P-Actions;

S-Actions → Action1 ; Action2 ; ...

P-Actions → Action1 // Action2 // ...

Client-server → From AgentId;
                To AgentId;

Multicast → From AgentId;
            To AgentId & AgentId & ...;
```

The model described here can easily be extended by adding new operators, properties, and concepts. Let us explain the way it considers the concurrency control. Locks are used to order transactions that access the same data items (DItem). The portion of the data items to which access must be serialized should be as small as possible for more concurrency.

Two types of locks are used: read locks and write locks. Before a transaction's read operation (RO) (resp. write operation (WO)) is performed, the server agent attempts to set a read lock (resp. a write lock) on the data item. As pairs of read operations from different transactions do not conflict, all the transactions reading the same data item share its read lock. So, read locks are sometimes called shared locks. It is not the case with a pair of read-write operations or write-write operations. These ones lead to conflicts with read-write locking (LRW) and write-write locking (LWW).

A lock manager is responsible for maintaining a table of locks for the data items of an agent. The lock manager provides the operations *lock* for requesting locks and *unlock* for releasing them. The lock manager is included in the CM which, from the specification of the lock primitive and the lock compatibility table, determines the conflicts and trigger the following *Locking-Algorithm* [5]:

```
1. When an operation accesses a data item within a
transaction T1:
   a. If the data item has a conflicting lock set by T2,
      then T2 waits until it is unlocked.
   b. If the data item has a non-conflicting lock set by
      T2,
      then the lock is shared and the operation proceeds.
2. When T1 is committed or aborted, all data items locked
for T1 are unlocked.
```

As the use of locks can lead to deadlock with transactions waiting for each other to release locks, the CM must trigger the *Deadlock-Algorithm* which uses lock timeouts:

```
1. Each lock is given a limited period t in which it is
invulnerable.
2. After t, a lock becomes vulnerable.
   a. If no transaction competing for the locked item
      then an item with a vulnerable lock remains locked.
   b. If any transaction is waiting
      then the vulnerable lock is broken.
```

5 Global Distributed Architecture

The architecture is based on a set of Communication Managers, each CM manages the interactions of a set of remote agents, detects conflicts and trigger the ad-hoc algorithms. From the specification of the metaknowledge in the CM and its relationship with the agent threads, the CM is able to construct *an interaction graph.*

An agent has four major functional components:
- A local database.
- A local knowledge source.
- A reasoning.
- A communication interface.

The global architecture is described in figure 4.

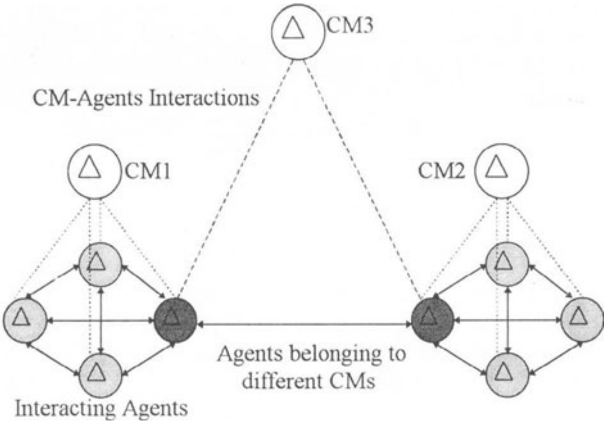


Fig. 4. Distributed Architecture

In this architecture, the CM role is to manage interactions between the agents of a particular group and to ensure its local consistency. The global consistency is the task of the higher-level CM which manages communications between agents related to different CMs (see CM3 in figure 4). We obtain a hierarchical architecture.

6 Interaction Graph Construction

The basic elements of an interaction graph are as follows: the agents are represented as points, and the communication links between these agents are represented as arrows between these points. This representation gives a graph of points and arrows, called an Interaction Graph. Communication relationships take two values: *S* or *R*. *Sij* denotes sending message from agent *i* to agent *j*, *Rij* denotes receiving message from agent *i* to agent *j*.

In the figure 5, we give an example of an interaction graph between agents A1, ..., A5.

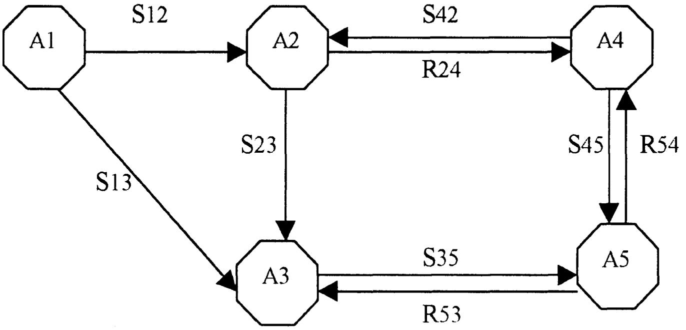


Fig. 5. Agent Interaction graph

Each arrow defines the interaction nature: sending or receiving message, the message type, the operation performed: reading or writing, the group of data concerned by the operation, the transmission mode, ...

For example, the agents A3 and A4 perform sending actions to the agent A5: *S35* and *S45*. The communication manager checks the message types:

- If the sending actions consist in sending a query to read some data in A5, the interaction will be interpreted as two read operations that could be performed concurrently.
- If the sending actions consist in reading a data item in the *S35* interaction and updating the same data item in the *S45* interaction, the CM detects the read/write conflict and runs the corresponding algorithm.
- If the sending actions consist in writing a data item in the *S35* interaction and updating the same data item in the *S45* interaction, the CM detects the write/write conflict and runs the corresponding mechanisms.
- If A3 sends a synchronous query to A5 (*S35*) and A3 receives synchronous queries from A1 and A2 (*S13* and *S23*), A1, A2, and A3 are blocked until the receiving actions are performed. The system performance is then reduced.

So, the asynchronous mode of transmission is strongly recommended in distributed multiagent systems in which agents are autonomous and scarcely dependent on some others. Therefore, the complexity of some applications and the increasing number of agents make possible the synchronous mode of transmission which may produce conflicts. The CM has to observe permanently the growing of the interaction graph.

7 Discussion and Future Work

We presented in this paper our preliminary efforts to specify the metaknowledge that the communication manager must hold in a DMS. We focused on the semantics of interactions between agents and the need of conflict detection and resolution. The study presented here outlined the features required by a multiagent system in which agents are physically distributed and have to cooperate to give a global solution. We are developing a DMS platform in which agents are implemented in ATOME-C++. Communication protocols between agents will be developed using RPC.

In a distributed multiagent system, agents cooperate, negotiate, interact following some social laws, make decisions, ... but they are discharged from resolving communication problems. An agent has to deal with its task and does not care about the communication mechanisms. So, with the design of a high level CM, we still have a multiagent system.

However, there are still some open questions on *how* to integrate the metaknowledge in the communication manager agents, *which* tool is needed to implement mixed communication model, *how* does the interaction complexity increase with the number of participating agents, and *what* is the implication of the system's ability to build a history during communication.

References

1. A. Attoui, “ Les Systèmes Multi-Agents et le Temps Réel ”, Eyrolles, 1997.

2. L. Berger, A-M. Dery, M. Fornarino and O. Jautzy, “ *First Steps to an Interaction and Communication Manager between Remote Objects*”, In Proc. ECOOP'97: European Conf. on Object-Oriented Programming, Finlande, June 1997.

3. B. Chaib-Draa and B. Moulin, " *Trends in Distributed Artificial Intelligence*", In Artificial Intelligence Review Vol. 6, 35-66, Kluwer Academic Publishers, 1992.

4. J-L. Chen, S. Lin and M-Y. Lee, " *Distributed Fault Management over ATM Networks* ", In Proc. IEEE FTDCS'97: Future Trends of Distributed Computing Systems, Tunis, Tunisia, pp. 196-201, October 1997.

5. G. Coulouris, J. Dollimore and T. Kindberg, " *Distributed Systems: Concepts and Design* ", pp. 34-41, 378-391, Addison-Wesley, 1994.

6. A. S. Cromarty, " *Control of Processes by Communication over Ports as a Paradigm for Distributed Knowledge-Based System Design*", In Readings in Distributed Artificial Intelligence, pp. 408-420. Morgan Kaufmann, California, 1988.

7. L. Gasser, C. Braganza and N. Herman, " *Implementing Distributed AI Systems Using MACE*", In Readings in Distributed Artificial Intelligence, pp. 445-450. Morgan Kaufmann, California, 1988.

8. V. R. Lesser and D. D. Corkill, " *Distributed Problem Solving Networks*", pp. 245-251, John Wiley and Sons, New York, 1987.

9. M. Mock, M. Gergeleit and E. Nett, "*Cooperative Concurrency Control on the Web* ", In Proc. IEEE FTDCS'97: Future Trends of Distributed Computing Systems, Tunis, Tunisia, pp. 118-123, October 1997.

10. D. D. Roure, L. Carr, W. Hall and G. Hill, " *Enhancing the Distributed Link Service for Multimedia and Collaboration* ", In Proc. IEEE TDCS'97: Future Trends of Distributed Computing Systems, Tunis, Tunisia, pp330-335, October 1997

11. R. G. Smith, " *The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver*", In Readings in Distributed Artificial Intelligence, pp357-366. Morgan Kaufmann, California, 1988.

12. J. D. Yang, M. N. Huhns and L. M. Stephens, "*An Architecture for Control and Communications in Distributed Artificial Intelligence Systems*", In Readings in Distributed Artificial Intelligence, pp. 606-616. Morgan Kaufmann, California, 1988.

Evolving the Scale of Genetic Search

John R. Podlena

Centre for Intelligent Systems
School of Biophysical Sciences and Electrical Engineering
Swinburne University of Technology, Victoria, Australia
jpodlena@swin.edu.au

Abstract. The Genetic Algorithm often has difficulties solving problems in which the scale of important regions in the search space (and thus the type of scale needed for successful search) differs. An algorithm is proposed in which the encoding precision for real based chromosomal structures is evolved concurrently with the solution, allowing the Genetic Algorithm to change the scale of its search to suit the current environment. The Algorithm is tested on three standard Genetic Algorithm test functions, and a cardboard box manufacturing application.

1 Introduction

The genetic algorithm is a paradigm which is based on the Darwinian "survival of the fittest" theory; a group of individuals in some environment have a higher probability to reproduce if their fitness is high. Offspring are created via a crossover operation (as in gene recombination) and mutation. The general idea is that the population over a number of generations will move towards a higher degree of fit individuals (algorithm outlined by Holland [1]).

The Scale Evolving Genetic Algorithm (SEGA) aims to exploit the fluctuating scale often found in the search space of optimisation problems [2]. This fluctuating scale can often lead to slower progress of the Genetic Algorithm in the initial stages of a search, and an inability to scale down a search when the algorithm is near an optimal value. This is particularly true of real number based encoding schemes [3].

The algorithm operates by encoding an individual chromosomes precision (for example, in the number of decimal places used). Those individuals within similar precision ranges are biased toward breeding with similar individuals, while individuals with precision ranges of largely different values are biased against breeding together.

The fundamental idea behind such a proposal is that the Genetic Algorithm operates on different scales, the scales being evolved based on performance at any given iteration. This effectively means that if the problem being solved needs a broad search initially, and a much finer search when the genetic population is close to the optimal solution, the population can change its precision to suit that situation during operation. It also means that the algorithm is less likely to become trapped in local minima due to the ability of the algorithm to produce a subset of the population with much broader mutation capability.

The idea of scaling chromosomal values seems to have been restricted to binary encoding schemes alone. Maniezzo [4] derived a scheme for a neural network / Genetic Algorithm hybrid which began with a reduction in the number of bits of representation per network weight value in the GENITOR [5] algorithm and increased the number of bits as iterations progressed. This was facilitated to combat what he saw as the negative effects of having a large/complex chromosome. Effectively it meant that the resolution of the weights was increased as generational time progressed, thus balancing the Genetic Algorithms broad search of weight values at first with the need for detail when a network was near convergence. Successful results using the algorithm were produced on simple Boolean functions. It can be seen, however, that the reduction of precision though a direct relationship with Genetic Algorithm iterations ignores the problems of local minima, and thus the possibility of premature convergence.

The algorithm also borrows ideas from the "Grey Scaling" mechanisms often applied to the binary encoded version of the Genetic Algorithm. Grey Scaling ensures that each possible value in a chromosome is a fixed hamming distance from each other. The theory is that this causes a less noisy distribution of possible solutions. Although this feature cannot possibly be fully catered for in a real valued encoding, some of its features can be reproduced if the distribution of possible solutions can be restricted through the use of sets of real valued numbers with the same precision.

The following paper outlines preliminary work into the use of SEGA. It includes the encoding schemes possible, the neighbourhood restriction, and the algorithms performance on three test optimisation problems and a real world application of the Genetic Algorithm to a cardboard box production problem (waste minimisation).

2 The Encoding Scheme and Algorithm

The Chromosomal encoding of the Algorithm is essentially simple. Given a set of real valued parameters, **R**, which make up the chromosome C,

$\mathbf{C} = \mathbf{Ri} \in \Re$, where $\mathbf{i} = \mathbf{1..[Chromosomal\ Size]}$

Each parameter has an associated precision parameter:

$\mathbf{C} = \mathbf{(R,P)}$ where $\mathbf{Pi} \in \mathbf{S}$

Set **S** is comprised of a fixed number of different precision values. For example, the **S** set could encode decimal precision:

$\mathbf{S=\{0.0,\ 0.1,\ 0.01,\ 0.001,\ 0.0001\}}$

or integer precision:

$\mathbf{S=\{1,\ 2,\ 5,\ 10,\ 50,\ 100\}}$

The algorithm follows the standard Genetic Algorithm, but sets all individual parameter values (rounding up based on precision values) directly before the cost evaluation:

```
UNTIL THE ERROR OF BEST INDIVIDUAL IS BELOW SOME MINIMAL VALUE:

  – set all individuals component parameter values based on respective precision
    values (rounding up)
  – calculate fitness for each individual in the population
  – calculate probability for parenting new generation
  – set n worst individuals to zero mating probability
  – copy m of the best individuals into new population (elitism)
  – breed individuals to fill next generation pool

REPEAT
```

Additionally, to ensure that the search algorithm does not produce a population which moves from a large precision average to a low precision average (thus losing some important solution detail), a breeding bias is placed on each new pair of selected parents. A probability, **Pr**, is calculated, which if not satisfied, disallows the breeding event between these two parents from occurring. This **Pr** can be calculated, for example, through the difference between the two parents average precision values in the precision set **S**:

Pr = k/[abs($\sum$ Positional Difference between P1i and P2i in set S) / Chromosomal Size]
where **P1i** and **P2i** are the precision values for parent 1 and 2,
and **i=1 .. [Chromosomal Size]**, and **k** is a constant (usually set to 1).

Note: In the case where the average difference was zero, or **Pr** was evaluated to >1, **Pr** was set to 1.

It was envisaged that this breeding bias would create neighbourhoods or "niches" [6] within the population, distributed across all possible precision values, giving the algorithm the ability to work on a number of different levels, but still allowing the migration of information from, for example, a high precision individual to a lower precision individual.

3 Three Function Optimisation Test Problems

Three function optimisation test problems were used on the algorithm. These were the Rozenbrock equation, F6 and F7 [7], which prove very difficult for hill climbing algorithms, other local heuristics such as the Tabu search [8] and also difficult for a genetic algorithm with a restricted mutation radius. To successfully find the optimal solution for the test functions F6 and F7, the standard genetic algorithm becomes very reliant on the mutation operator. To further handicap the initial population, the "random" selection of individuals was permitted only to be outside a certain radius of the optimal solution point. The following functions optimal solutions were moved away from the origin to preclude the trivial solution of setting the precision to zero.

The functions tested were:

Rozenbrock: $100\left(x-y^2\right)^2+(y-1)^2$

F6 : $0.5+\frac{\sin^2\sqrt{x^2+y^2}-0.5}{[1.0+0.001(x^2+y^2)]^2}$

F7: $(x^2+y^2)^{0.25}\left[\sin^2(50(x^2+y^2)^{0.1})+1.0\right]$

4 A Cardboard Box Cutting Optimisation Problem

An optimisation problem exists in the cardboard box manufacturing industry in the cutting of cardboard blanks (from which the cardboard box is created through folding and gluing) from large rolls of cardboard. This problem is essentially one of waste reduction. "Patterns" must be devised which cut out the maximum number of box blanks of various sizes with the least amount of wasted cardboard (see figure 1).

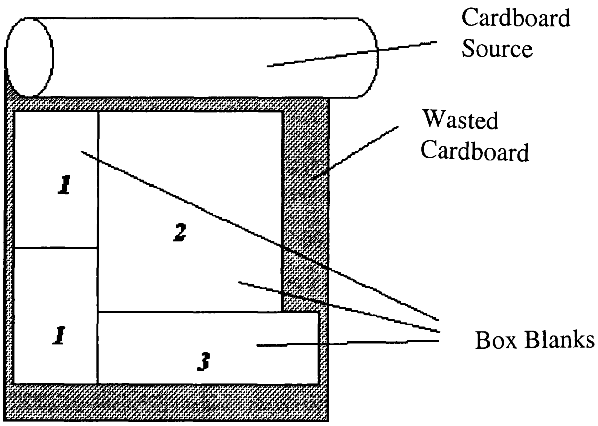


Fig. 1. Three types of cardboard box blanks have been cut out in this example. It can be seen that this particular "pattern" has its own cost in terms of wasted cardboard

The existing algorithms for solving such problems are often based on simple sets of rules to find a solution which, although not often near optimal, are accepted due to a lack of alternatives. One alternative to the use of such algorithms is to use the Genetic Algorithm to breed solutions consisting of a number of patterns to be used for a set of given box types and quantities. Such an application of the Genetic Algorithm was produced and is outlined below. The application provided an excellent opportunity to test SEGA on a real world application.

5 Chromosomal Representation and Decoding

To ensure that all legitimate pattern types were able to be represented in the chromosome, it was decided that the chromosome should consist of a maximum number of patterns, each with a maximum number of possible box types (taken from a fixed order list).

For example, the following could be the input order list:

Box Type	*Length (mm)*	*Width (mm)*	*Quantity*
1	1222	1222	1200
2	1333	1333	30000
3	1444	1111	4000
4	1111	1666	200
5	2000	300	4000

from which the following chromosome could be created, consisting of four patterns with four orders in each pattern:

{Pattern1:{order1,order3,order5,order1},
Pattern2:{order3,order2,order5,order1},
Pattern3:{order1,order3,order4,order5},
Pattern4:{order3,order4,order2,order2}}

Each order within the chromosome was initialised with random selections of orders, and each order within the chromosome was given a random quantity (within the range of the original order). The decoding was done in such a way as to prevent any individual chromosome from being invalid. This was done to overcome the sometimes debilitating effects on genetic search of removing nonviable individuals from the population, or by penalising such individuals [9]. The decoding worked in the following manner:

```
while all patterns not processed, loop through patterns
  while this pattern is not complete, loop through orders in this pattern
    If this orders dimensions can fit into current partial pattern solution,
    And the quantity of the order has not already been completed Then
    place this order into the partial pattern solution.
    EndIf
  EndLoop
EndLoop
```

The decoding algorithm continued in this way, producing a set of patterns which together complete all orders.

The cost function for each individual was calculated as:

Cost = Trim% x A + TotalLength x B

where Trim% is the total percentage waste from the solution, TotalSolutionLength is the total length of the solution, and A and B are constants set by relative importance.

For the breeding of two individuals, crossover was applied on a pattern and order level, with mutation occurring through the random replacement of orders within an individual with randomly selected orders. A mutation of the quantities within each order in an individual was also implemented. It was on this later variable, the order quantity, that the SEGA was applied.

6 Results

Table 1 outlines the results of SEGA on the three function optimisation problems, comparing SEGA with the standard Genetic Algorithm approach, and with the SEGA with the neighbourhoods / niches scheme. All averages were taken over 1000 runs, each run with the same initial population for the three different methods (i.e. 1000 different initial populations).

Table 1: Average Iterations.	Ave. It.s no SEGA	Ave. It.s SEGA	Ave. It.s for niche SEGA
Rozenbrock	395.63	75.12	72.72
F6	124.30	96.74	92.61
F7	1027.21	73.26	70.43

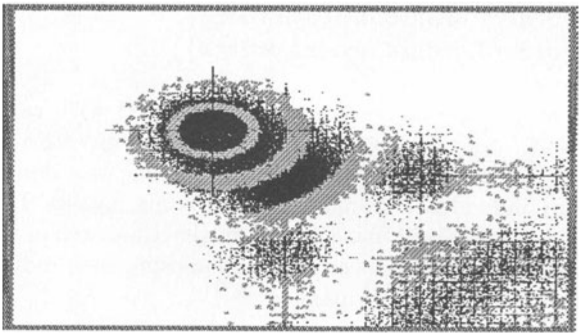


Fig. 2. This diagram shows an amalgam of a number of runs of the SEGA on the F7 optimisation problem. The light grey regions indicate low fitness regions, the black points represent individual evaluations. Note that two precision values (namely 0.1 and 0.01) can be clearly seen as square distribution lines.

After 100 trials using the same initial population for the cutting algorithm with and without the SEGA (each run ran for 50 iterations), it was found that the SEGA version out-performed the standard Genetic Algorithm 71% of the time, as can be seen in table 2. Figure 3 displays the percentage of trials in which the SEGA converged quicker than the standard Genetic Algorithm on all four test problems.

Table 2: Sega Wins over Standard GA	Wins	Losses	Draws
Rozenbrock	1000	0	0
F6	902	91	7
F7	985	15	0
Cutting Algorithm	71	29	0

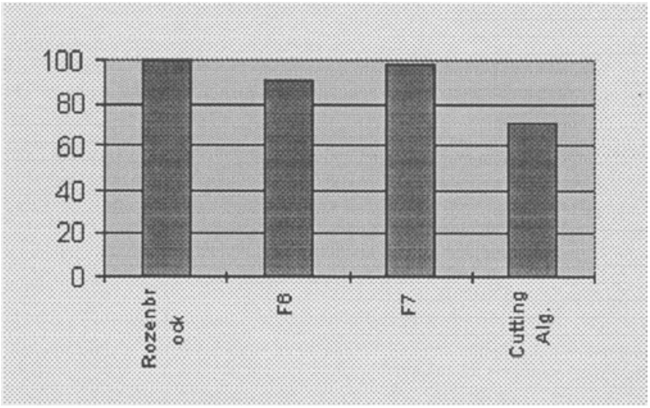


Fig. 3. This graph displays the number of trials in which the SEGA outperformed the standard Genetic Algorithm as a percentage of total trials.

7 Conclusions

In conclusion, it can be seen that the Scale Evolving Genetic Algorithm outperforms the standard Genetic Algorithm on both the function optimisation test problems, and on the cardboard cutting problem.

In the case of the function optimisation trials, the algorithm accelerated convergence by a factor of up to 10 times, with a consistent improvement over the standard Genetic Algorithm in 92-100% of the trials. The niche mechanism produced an improvement in convergence iterations over the SEGA without niches by around 4%.

In the case of the cardboard cutting problem, the SEGA was found to outperform the standard Genetic Algorithm in 71% of trials, on average having a fitness of 2% higher. Although these results are less striking, it is significant in the savings in its reduction of cardboard wastage in the production process.

Future work will include an expansion of the algorithm into a multiple population based distributed Genetic Algorithm. The migration will be primarily controlled via a similar mechanism to the niching process described above.

References

1. Holland, J. H.: Adaption in Natural and Artificial Systems, J. H. Holland, University of Michigan Press, Ann Arbor, MI, 1975.
2. Davis, L.: Bit-Climbing, Representational Bias, and Test Suite Design, Proceedings of the Fourth International Conference on Genetic Algorithms, pp 18-23, California, July 1991.
3. Goldberg, D.E.: Real-coded Genetic Algorithms, Virtual Alphabets, and Blocking, University of Illinois at Urbana-Champaign, Technical report No. 90001, September 1990.
4. Maniezzo, V.: Genetic Evolution of the Topology and Weight Distribution of Neural Networks, IEEE Transactions of Neural Networks, Vol. 5, No. 1, pp 39-53.,1994.
5. Whitely, D.: The GENITOR Algorithm and Selection Pressure: Why Rank-Based Allocation of Reproductive Trials is Best, ICGA 1989.
6. Deb, K., Goldberg, D.E.: An Investigation of Niche and Species Formation in Genetic Function Optimization, Proceedings of ICGA, 1989.
7. De Jong, K.A.: Analysis of the Behavior of a Class of Genetic Adaptive Systems, PhD Dissertation, Department of Computer and Communication Sciences, University of Michigan, Ann Arbor, MI, 1975.
8. Woodruff, D.L., Zemel, E.: Hashing Vectors for Tabu Search, Annals of Operations Research, Vol. 41, pp. 123-137, J.C. Baltzer AG, Switzerland,1993.
9. Michalewicz, Z., Michalewicz, M.: Pro-life Versus Pro-choice Strategies in Evolutionary Computation Techniques, in Computational Intelligence: A Dynamic System Perspective, Eds. Marimuthu Palaniswami, Yuanni Attikiouzel, Robert Marks, David Fogel, Toshio Fukuda, IEEE Press, NY, 1995, pp137-151.

Ensembles of Neural Networks for Analogue Problems

David Philpot and Tim Hendtlass

Centre for Intelligent Systems,
School of Biophysical Sciences and Electrical Engineering,
Swinburne University of Technology.
P.O. Box 218 Hawthorn 3122. Australia.
Email: dphilpot@swin.edu.au, tim@swin.edu.au.

Abstract. This paper discusses a technique that uses several different networks working together to find a relationship that approximates analogue training sets. Each network learns a different region of the training space and all these regions fit together, like pieces of a jigsaw puzzle, to cover the entire training space. This analogue approach is an extension to a technique previously developed for solving digital problems. The networks can be of any type (eg backprop, cascade). However, virtually any other technique can be used in place of the networks : evolved polynomials, DRS, Taboo search, Nearest Neighbour, and other statistical techniques.

1. Summary of Digital Technique

This summary discusses the original digital technique that involves using several neural networks working together to solve one digital problem [1]. This ensemble of neural networks is 'grown', and typically takes the form of a series or branch of individual networks. Each network learns a separate region of the training space, and all the regions from these networks fit together, like pieces of a jigsaw puzzle, to cover the entire training space.

All of the digital training sets are learned with 100% accuracy, providing the set has a many-to-one relationship between inputs and outputs. The stability and robustness of this technique has meant that much of the guess work that has traditionally been used to determine the various parameters such as architecture, learning coefficients, and so on, has been removed. Better parameters will often lead to a faster solution with fewer networks, but often they can be widely varied and a solution still found.

The individual networks do not have to be the same type. This means that the technique can take advantage of the many different architectures and learning rules that are available, choosing the best network for each of the different parts of the training space accordingly.

Test points are classified primarily on the learned (hyper)region they fall into. A technique for enhancing the generalisation capability of the ensemble in inter-region space, using relearning, is also described.

1.1. Training the Networks

The training process begins conventionally in that one network is trained on all of the training data until learning has stabilised. Once this has been achieved, only the points that were not learned by the first network are passed onto the second network for training. The second network then trains on these points until learning has stabilised, and only the unlearned points from the second network are passed onto a third network for training. This process continues until eventually all the training data has been learned. Each network learns a particular region of the training space and also records the range over which its outputs are valid. This range is divided into 'unlearned', 'learned ON' or 'learned OFF' regions (Fig. 1). The learned ranges consist of actual output values for which there is only one desired output. The unlearned range(s) consist of actual output values for which the desired output could be either of the desired outputs. These 'markers' (eg. A,B,C,D in Fig. 1) are stored with the network. The end result is a 'series' of networks whose combined relationship, using all the regions matches the training data exactly (Fig. 2).

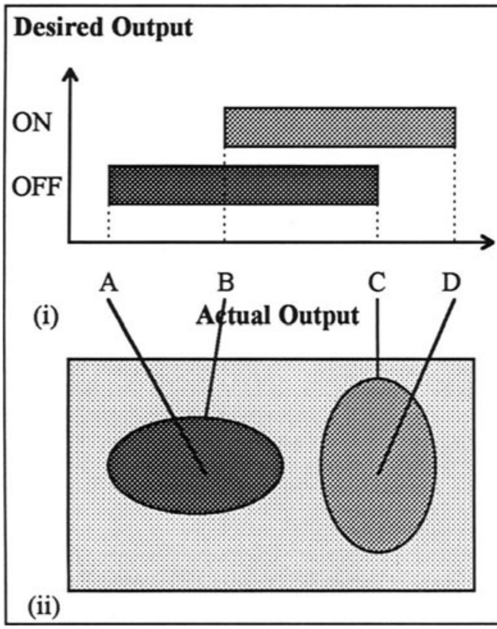


Fig. 1.
(i) This is a typical graph showing the relationship between desired output (ON or OFF) against the actual output from one network. The regions obtained from the actual output are :
A-D : Full output range.
A-C : Desired output OFF range.
B-D : Desired output ON range.
A-B : Learned desired OFF.
B-C : Unlearned range.
C-D : Learned desired ON.

(ii) Indicates how the ranges (i) can relate to a 2D training space.
▩ Learned region OFF (A-B)
□ Unlearned region (B-C).
▨ Learned region ON (C-D).

Sometimes the regions learned by different networks do not overlap. This can lead to testing points between these regions being incorrectly classified. These errors can often be reduced by relearning some of the points already learned by previous networks in addition to the unlearned points. The learned points that are passed onto later networks for relearning are those just inside the learned region that are also near to an unlearned region. Increasing the training set in this way helps ensure that all the problem space is learned and helps to prevent testing points falling into unknown regions.

The training technique can also be modified to produce a 'branch' of networks rather than a 'cascade' of networks (Fig. 3). A 'branch' of networks is produced when there is more than one unlearned region. The training points from each part are passed to separate networks. This may be advantageous when a training set contains several separable features. A 'branch' ensemble will sometimes consist of more networks than a 'series' of networks. Despite this, training may be faster because networks are often trained on fewer points, and the relationships in these points are simpler. This can lead to better network generalisation. A 'branch' of networks will often lead to faster testing than a 'series' of networks.

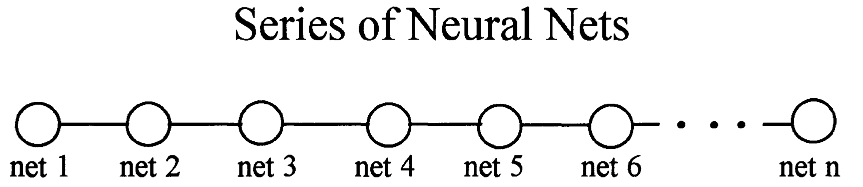


Fig. 2. Unlearned points from the first net get passed onto the second network to learn. This process continues until all the training points have been learned. The end result is a 'series' of neural nets.

1.2. Testing the Networks

When testing on a 'series' of networks, the inputs for the test point are presented to the first network that was trained. If the output for this network indicates that the test point falls in a 'learned' region (by comparing the output with the 'markers' stored with the network), then it is classified by the first network. If, however, the test point falls in an 'unlearned' region, it is passed onto the second network in the 'series' and the same procedure is repeated. The test point keeps moving down the 'series' of networks until one of the networks classifies it.

Testing on a 'branch' of networks is similar to a series. The main difference is that it depends on which unlearned region the outputs for the test point falls into as to which network at a branch point the test point is passed onto. The test point starts at the top of the branch ensemble with the first network and continues moving down one of the branches until it is classified. This often leads to faster testing because each branch path is often shorter than a full 'series' of networks. As a result, a test point does not need to be tested on as many networks before being classified.

2. Analogue Extension

The analogue extension allows data sets to be solved that contain analogue outputs (as opposed to digital outputs). It is a simple extension to the digital approach, but one in which the digital approach is used.

The first step involves training the first analogue network on all the training data until learning has stabilised. All the points that have been learned to within a desired accuracy are considered learned, the rest are passed onto later analogue networks for training. This process is repeated until all the training data has been learned to within a desired accuracy.

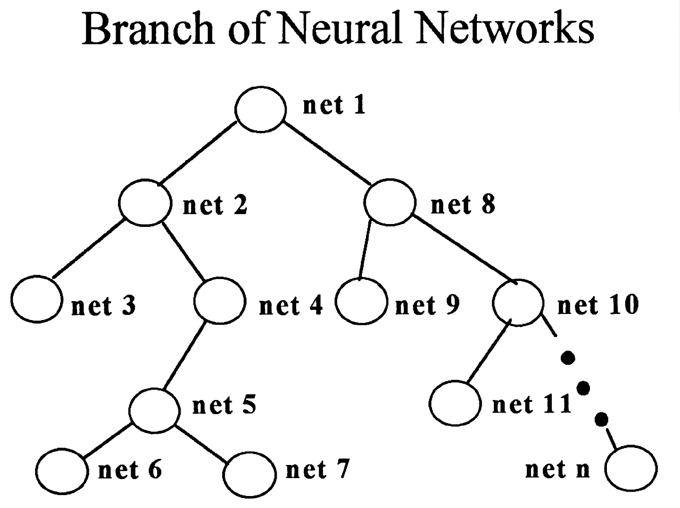


Fig. 3. - Unlearned regions can be split up and learned by different networks. The end result is a 'branch' of neural networks.

The main difference between the analogue and digital approaches lies in the way learned and unlearned regions are stored with the network. All that it is needed when using the digital approach are markers that map regions of the output for that network as either 'unlearned', 'learned ON' or 'learned OFF'. The analogue approach needs a more complex approach because the learned regions are analogue rather than simply ON or OFF. When training for an analogue network has completed, the learned and unlearned points are easily calculated by determining if the outputs are within a desired accuracy. A new training set is then generated that is made up of all the points in the training set for the analogue network however the outputs are now DIGITAL - ON if learned, OFF if unlearned. An ensemble of digital networks is then 'grown' to learn this digital training set. In other words the digital ensemble effectively maps which regions of the training space have been learned by the current analogue network. This ensemble is stored with the analogue network and is used to during testing to determine if a test point lies in a learned or unlearned region for the analogue network (see Fig. 4). If it lies in a learned region, then the current analogue network determines the output for the test point, if not it is passed to later analogue network.

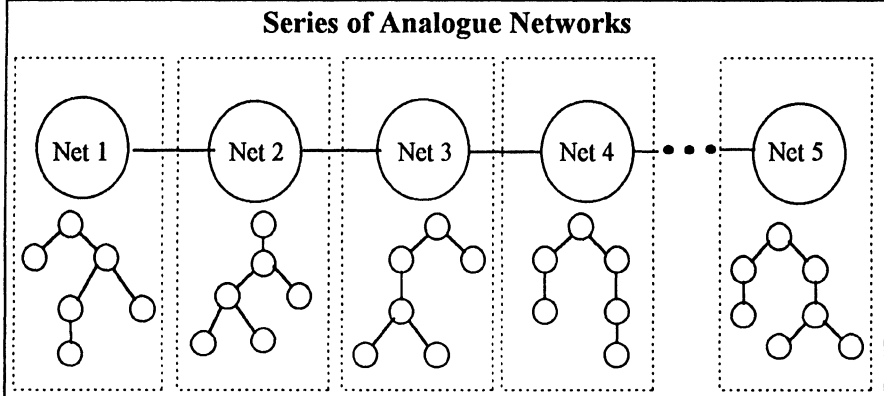


Fig. 4. A typical form that a series of networks would take for analogue data. For each analogue network there is an ensemble of digital networks that is used to map the learned and unlearned regions for the analogue network.

3. Results

The data set made up of 500 points describes the distance travelled by a projectile from the starting point when launched with different starting conditions. The data set has two inputs and one output describing the relationship between initial velocity u (input), angle θ (input) and distance travelled from the starting point d (output) of a projectile. The angle θ varies from 0 to 1.57 rads (0 to 90 degrees), and the initial velocity from 0 to 20m/s. The data set was generated using equation 1.

$$d = u^2 \cos(\theta) \sin(\theta) / 5 \qquad (0 \le u < 20,\ 0 \le \theta < 90) \qquad (1)$$

The training set was solved with a desired training accuracy of 95%. The training accuracy is the accuracy that the output for a point must be within to be considered learned. This means that the output from the ensemble will not match the training data exactly (as with the digital approach), but it will be within the specified training accuracy.

The trajectory data set is represented as a contour map in Fig. 5(i). The greatest distance travelled occurs when the angle is 0.785 radians (45 degrees), and the initial velocity is at a maximum (20m/s).

The first network trained was *analogue* (which is always the case when dealing with analogue problems). It learned 490 out of the 500 points in the training set. A new training set was then created for the digital classification ensemble. This was made up of the same 500 points as the original data set, but the outputs are now *digital.* The 490 points that were learned were given the digital output of high, and the other 10 unlearned points low. It then took four digital networks (networks 2-5) to learn this training set.

Training now begins on the second *analogue* network (network 6). There are now only 10 points in this training set (the 10 unlearned points from the first analogue network). This network now learns 6 of these points which takes the total points learned from the original analogue set to 496. The 10 points are then used to create another digital training set for the digital classification networks (6 high, 4 low). It then took only one digital network (net 7) to learn this *digital* training set.

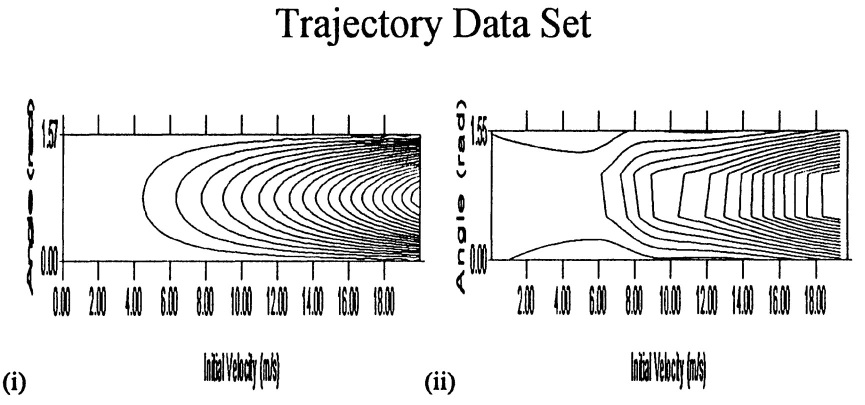


Fig. 5. Contour maps of (i) the training data and (ii) the generalised ensemble output. The distance travelled by a projectile (output) depends on the initial velocity (x axis) and the angle (y axis). The maximum distance travelled occurs at maximum initial velocity (20 m/s) with angle 0.785 rads (45 degrees).

Network 8 was the next *analogue* network to be trained on the remaining 4 points (unlearned from net 6) which learned all of them. One *digital* network (net 9) was required to learn the training set made up of 4 high and 0 low points (all points have been learned). It usually only takes one digital network after the last analogue network since all the points have been learned and are thus all high which is a relatively trivial problem to solve. The actual structure of the ensembles created (analogue networks and their digital classification ensembles) is shown in Fig. 6.

The testing accuracy for this ensemble was 97.0% which is the percentage of the testing data that gave outputs within an accuracy of 95% (training accuracy) when passed through the ensemble.

4. Conclusion

This paper showed how to 'grow' an ensemble of networks to solve analogue data sets by making a couple of simple extensions to the digital approach. The training set used was a relatively trivial trajectory problem but does show that the technique works. There is still much research that needs to be done before the power, and limitations, of this approach are better understood.

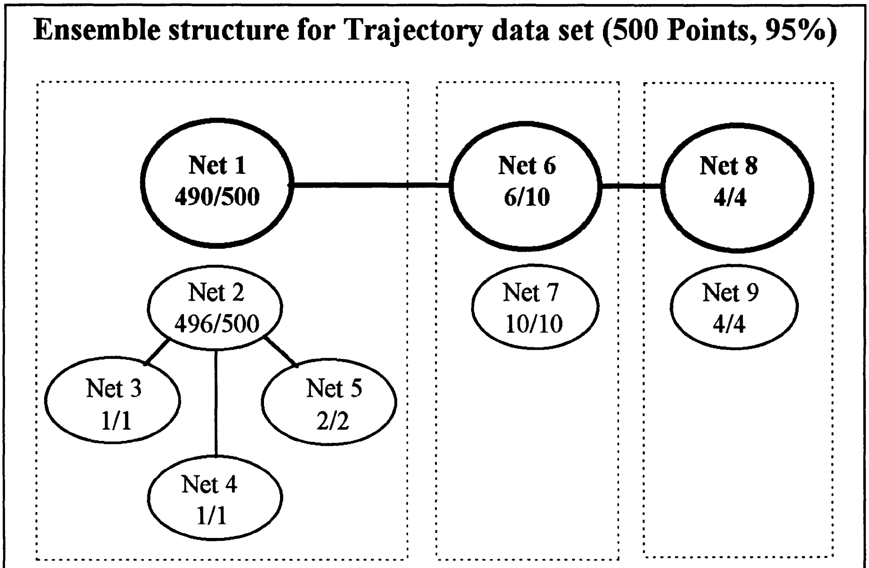


Fig. 6. This ensemble structure is the solution found for the analogue trajectory data set with 500 points solved with a training accuracy of 95%. There are three analogue networks in series (nets 1,6,8) with their corresponding digital classification ensembles below them. The fraction associated with each network shows the 'points learned by the network'/'training points for the network'

References

1. Philpot, D., Hendtlass, T. (1997) *Ensembles of Networks for Digital Problems.* Proceedings of ICANNGA'97 (in press).

An Evolutionary Algorithm with a Genetic Encoding Scheme

Howard Copland and Tim Hendtlass

Centre for Intelligent Systems
School of Biophysical Sciences and Electrical Engineering
Swinburne University of Technology
Hawthorn 3121,
Victoria, Australia
e-mail: hcopland@swin.edu.au or thendtlass@swin.edu.au

Abstract. This paper explores a variant of an evolutionary algorithm which has no explicit parameter representation, but rather represents parameters across a common "genetic string" in a manner much more akin to the storage of genetic information in DNA. Results for several common test functions are given, and compared with the performances for a standard evolutionary algorithm.

1 Introduction

The typical chromosome used in an evolutionary algorithm is a potential solution that consists of *genes* which describe individual parameters. For example, a chromosome describing an artificial neural network may have genes in the chromosome for weights, learning rates, transfer functions, and so on. These are all discrete, with a change in one of the parameters having no direct impact on any other parameter.

Biological chromosomes are vastly different to this in structure. In organisms, many genes, which are coded from DNA bases, commonly interact to produce a single trait. Conversely, a single gene may contribute to several different traits. At times, entirely different sets of genes could produce the same characteristic. From the genotype[1] alone it is difficult, if not impossible, to say what a mature organism's phenotype[2] will be. Any single point of genetic information interacts with (potentially) many other points. Not all genes are expressed, and much DNA in chromosomes has no known purpose. DNA sequences which are expressed are termed exons and those which are not expressed are termed introns. The structure of the genome is dynamic, in many organisms substantial sections of DNA called transposons change location in the genome.

Little of this applies to the conventional evolutionary algorithm as it deals with already expressed characteristics predetermined by the user. In simple

[1] Genotype is defined as the genetic constitution of an individual, the sum total of its genes where genes are the basic unit of biological information.

[2] Phenotype is the sum total of observable features in an individual, regarded as the result of the interaction of its genotype with the environment.

terms, the mapping of genotype to phenotype is a far more complex phenomenon in organisms than in the solutions sought by evolutionary algorithms. A typical artificial chromosome contains hereditary information about the phenotype which is a solution. That such descriptors rather than some genetic structure is used is perfectly sensible. That genetic-style operators are then applied seems less so. Charles Darwin, after all, knew nothing of molecular biology.

2 Coding of Solution Parameters

To determine if there is advantage to a genetic-style representation and the subsequent application of genetic-style operators within the framework of an evolutionary algorithm, an *attribute chromosome* is coevolved in conjunction with a *control chromosome*, which modulates the output of the former.

An end solution consists of multiple chromosomes, the minimum being one from each of the attribute chromosome and control chromosome populations; the maximum number of control chromosomes is equal to the number of parameters in the solution. The attribute chromosomes are long binary arrays and the control chromosomes are much shorter arrays of real numbers.

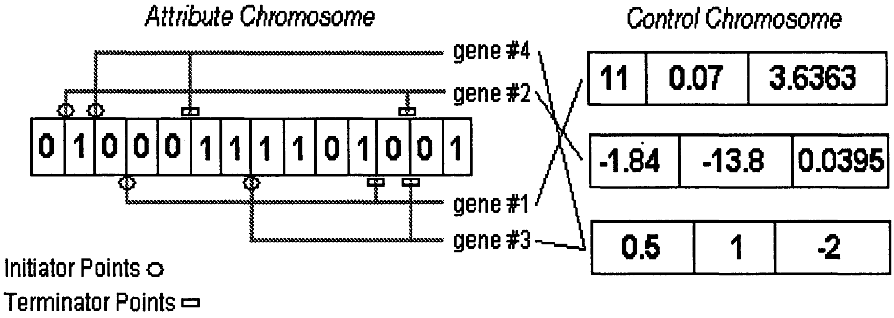
A solution parameter is constructed by selecting initiation and termination points randomly on the attribute chromosome. This specified section is termed a gene. This sequence is then summed, the 1s in the sequence being analogous to exons and the 0s analogous to introns. The expression of this integer is then modulated by the control chromosome. The control chromosome may be constructed in a number of ways, depending on the problem: a sum or multiplication of values, even as coefficients for a function. For simplicity, in the work described, a multiplication of values is used.

While this may appear an unwieldy means of arriving at a value, such a structure is more analogous to a biological chromosome, with control genes [1], [2] that modulate the expression of other genes. Neurons, muscle cells and skin cells, for example, all have the same DNA, but the expression of that DNA is different as a result of the different control genes involved.

The traditional encoding of, for example, a four parameter solution (each parameter represented by a single number) would be as follows:

14	6.0179	-3	0

The genetic representation of such a chromosome would look like the diagram below, which is considerably more complex. Note that genes #3 and #4 share the same control chromosome, and that all genes, to some extent, overlap.



The parameter values in both cases are the same. Take, for example, parameter two, which has the value of 6.0179. In the genetic representation, the attribute chromosome yields a value of 6, and the second control chromosome is used. Using multiplication of the control chromosome values, this yields a value of 1.002984, giving a final value of 6.0179.

Evolutionary algorithms are used where problems occupy complex hyperspaces, the interactions of the various factors involved making analysis difficult. Conventional evolutionary algorithms are a good approach to such problems; augmenting the evolutionary algorithm with a genetic analog offers several potential advantages.

The "holographic" nature of the genetic representation results in a significantly higher information density in the chromosomes, which is one of its key strengths. Any change in an attribute chromosome, by mutation for example, has immediate impact on all genes which share that base, as the parameters are encoded in a distributed fashion with a high probability of overlap. Crossover between individuals results in significant movement within the solution space, even if their fitnesses are nearly equal, as many genes are affected. The encoding scheme thus allows a more rapid exploration of the solution space. Consequently, the search path is quite different to that when the traditional encoding scheme is used. In terms of mutation, the traditional encoding scheme means a mutation "step" along one given dimension, whereas with genetic encoding a mutation results in a mutation "step" along many dimensions. With crossover, fitness based breeding in the traditional encoding scheme results in population "clusters" in the fittest found regions of the solution space, even if, at initialization, the distribution of individuals is more uniform. Such clustering is not intrinsic to the genetic encoding scheme, and so a more diverse distribution results.

In the work described, each attribute chromosome is 100 entries long, containing a random sequence of 1s and 0s. Each control chromosome is 5 entries long, each entry a real number. The number of attribute and control chromosomes in the population is 100 and 10, respectively.

All problems are surface minimization problems, where the goal is to evolve x and y coordinates resulting in the minimum value for z. The solution consists of two or three chromosomes, and a series of coordinates which specify the initial and terminal points for genes on each of the chromosomes. Two or three chromosomes could be involved as although the same attribute chromosome is used for both x and y, the control chromosomes for modulating the expression of each value could differ. Each x and y value is constructed as the product of the sum of the 1s in the attribute chromosome and the product of the real values in the control chromosome.

3 Breeding and Mutation

Due to the number of structures in the genetic algorithm, there exist several mutation types. First, the length of a gene on an attribute chromosome can change. Second, there may be exon/intron switching within the attribute chromosome; that is, a 1 could become a 0, or vice versa. Third, the control chromosome modulating the expression of the attribute chromosome could change. Finally, the values within the control chromosome could themselves mutate. In the work described, the probabilities for these mutations were 0.05, 0.025, 0.0125 and 0.05, respectively, though these values are not critical. Additional constraints on mutation concerned the magnitude of the mutation. Gene length mutation could equal but not exceed 10% of the chromosome length, and mutation to the values within the control chromosome could equal but not exceed either 1% or 10% of the original value, as specified.

Crossover is uniform for all attribute chromosomes, though this is complicated by the transposon analog. During attribute chromosome breeding, a parent is chosen at random for each of the new individuals and 20% of the parent chromosomes sequence is copied intact directly into the offspring. The beginning point of this is chosen at random, but it could not begin less than 20% of the chromosomes length away from the end of the chromosome; wrap around is not permitted.

Breeding probability is based on the overall solution fitness for attribute chromosomes. Control chromosomes, however, were considered to be of uniform fitness, and the number of parents any individual offspring might have is not limited to two, but limited to the number of entries in the control chromosome. Each could potentially be from a different parent.

The initialization procedure for attribute chromosomes carried a 0.5 probability that any given entry is a 1. The initiation point for each gene is a point randomly determined to be in the first half of the attribute chromosome, and the termination point for each gene is a point randomly determined to be in the second half of the attribute chromosome. The control chromosome associated with each gene in each attribute chromosome is then randomly determined.

The four test functions [3] are as follows:

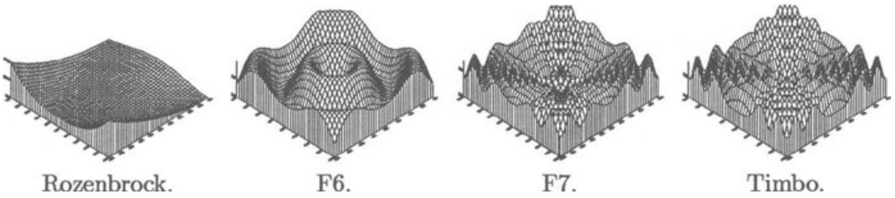

Rozenbrock. F6. F7. Timbo.

Fig. 1. Test Surfaces

The equations for the above surfaces are as follows, left to right:

$z = 100(x - y^2)^2 + (y - 1)^2$

$z = 0.5 + \frac{\sin^2 \sqrt{(x^2+y^2)} - 0.5}{(1+0.001(x^2+y^2))^2}$

$z = (x^2 + y^2)^{0.25}(\sin^2(50(x^2 + y^2)^{0.1}) + 1)$

$z = 1 + (x^2 + y^2)^{0.25} - \cos(5\pi\sqrt{x^2 + y^2})$

The standard evolutionary algorithm uses a two element chromosome, elements corresponding to the x and y values. Breeding probability is fitness based. Mutation probability is set at 0.05 and the mutation radius at 1% or 10% as specified. The population size is set at 100. At initialization, the values within the chromosomes are within the range ±10. Elitism is present in both the standard evolutionary algorithm and the evolutionary algorithm augmented with the genetic encoding scheme, with the single best solution preserved intact into the next generation.

4 Results

Tables 1a and 1b compare the performance of a standard evolutionary algorithm with that of the evolutionary algorithm augmented with the genetic encoding scheme (the "genetic algorithm"). Note that if the standard evolutionary algorithm failed to converge within 50,000 generations then that attempt to evolve a solution is deemed unsuccessful and the number of generations not included in the average. The results are the averages for 100 trials of each problem. Minimization is to within 0.01 or 0.05 of the true minimum, as stated.

Comparisons between the evolutionary algorithm and the genetic algorithm were not easily made, as the degree of freedom is greater for the latter as more structures are evolved. Nevertheless, the closest "match" between the two is decided to be where the mutation parameters for the genetic control chromosome and the standard chromosome are identical.

In the tables below, *#NC* refers to the number of trials which are not converged at 50,000 generations for the standard evolutionary algorithm. The (2,2) notation on F7 and Timbo refers to an offset applied such that the minimum is not located at the (x,y) coordinates of (0,0) but at (2,2).

Table 1. Comparison of Evolutionary and Genetic Algorithms, Convergence Criterion 0.01.

	Evolutionary Algorithm				*Genetic Algorithm*		
Problem	**Max**	**Min**	**Mean**	**#NC**	**Max**	**Min**	**Mean**
Rozenbrock	1,698	1	353	0	63,904	65	8,313
F6	19	1	4	0	80	7	20
F7(0,0)	48,492	331	9,753	29	3,218	160	1,420
F7(2,2)	49,196	8,330	26,673	75	33,107	1,515	8,707
Timbo(0,0)	6,822	360	2,653	0	10,815	1	512
Timbo(2,2)	12,109	322	3,230	0	13,375	52	3,099

Table 1a - mutation parameters: 0.05 probability, 1% radius.

	Evolutionary Algorithm				*Genetic Algorithm*		
Problem	**Max**	**Min**	**Mean**	**#NC**	**Max**	**Min**	**Mean**
Rozenbrock	97	3	25	0	80,273	114	8,955
F6	31	1	6	0	20	2	5
F7(0,0)	49,992	3,683	19,055	78	12,179	75	4,073
F7(2,2)	47,537	975	11,930	28	6,855	604	2,846
Timbo(0,0)	44,656	1,983	19,723	2	12,146	28	2,185
Timbo(2,2)	49,858	1,422	21,688	17	12,809	353	2,498

Table 1b - mutation parameters: 0.05 probability, 10% radius.

Table 2. Genetic Algorithm Comparisons for 0.01 and 0.05 Thresholds on Rozenbrock.

Threshold	**Max**	**Min**	**Mean**
0.01	63,904	65	8,313
0.05	5,016	228	1,018

Mutation Parameters: 0.05 probability, 1% radius.

The genetic encoding scheme results in a general improvement over the standard representation for the two functions considered more demanding: F7 and Timbo; problems seen to traditionally rely largely upon creep mutation [4] at the end of the evolutionary run in order to achieve their minimums. The search path with the genetic encoding scheme, with its multidimensional stepping and more diverse distribution, bypasses the incremental mutation of the traditional encoding scheme.

However, this does not mean the genetic algorithm results in simply a superior means of creep mutation as indicated by a comparison between the results for mutation radii of 1% and 10%. A greater mutation range is generally not advantageous when creep mutation is required, and the advantage of the genetic algorithm is maintained (e.g. F7(0,0), Timbo(0,0)) to a great extent, even enhanced (e.g. F6, Timbo(2,2)), when the mutation radius increased. This increase in mutation radius results in significantly poorer performances by the evolutionary algorithm for F6 and both versions of F7 and Timbo.

Shown most clearly for Rozenbrock, the genetic algorithm has difficulty in achieving the minimum when the minimization is of a comparatively simple surface. While the genetic encoding scheme places the solutions in the right area rapidly (see Table 2), relatively speaking, the fine tuning to achieve the solution takes considerably longer.

The fitness (as opposed to rank) based breeding scheme used with the genetic algorithm compromises its ability to converge rapidly when presented with the Rozenbrock function. The populations used are, for all problems, subject to the same initialization. This results in initial fitness typical within the following ranges:

a. Rozenbrock[3] 1.0x10^5 - 1.5x10^{12}.

b. F6 0.5 - 2.0.

c. F7 5 - 30.

d. Timbo 2 - 20.

With a breeding probability based on fitness, the poor fitness of the population for Rozenbrock reduces the effective breeding population; in place of a population of 100 individuals, there are perhaps 10 individuals contributing to the breeding pool and the best of these solutions is extremely poor.

Also, it is likely that the genetic algorithm will be generally poorer on gradient-descent type surfaces due to the highly non-linear nature of inheritance. Offspring are not necessarily situated near parents in the feature space.

The advantage of the genetic algorithm evident in F7 and Timbo could be, to some extent, a false advantage. The global minimum and the solution values for x and y both correspond to zero, and the genetic algorithm can find zero points quite rapidly. All that is required is for a gene to be a sequence of zeros in the attribute chromosome, which can be achieved through crossover, mutation of gene lengths or exon/intron switching. The gene thus expressed is "off" and so has no value a zero. The advantage here for F7 and Timbo is most pronounced where the mutation radius is 1% for the control chromosome. The greater stability of the control chromosome ensures the bulk of changes occur in attribute chromosome, so individuals are selected for shorter genes and frequency of introns.

To ascertain if the genetic algorithms tendency to readily evolve zero values is the reason for the improved performance, F7 and Timbo are offset by (2,2): the multidimensional searching ability of the genetic algorithm is being evaluated. For the 1% mutation radius, the evolutionary algorithm has a mean increase of 3.6 fold in generations to convergence for F7(2,2) compared to F7(0,0). The corresponding figure for the genetic algorithm is an increase of 6.2 fold. For Timbo(0,0) function, the increases are 1.2 fold and 6.1 fold for the evolutionary and genetic algorithms, respectively. The increases in the genetic algorithms

[3] The nature of the Rozenbrock function accounts for the extremely poor fitness at initialization. Away from the minimum, the function rapidly climbs to high values over a very short range.

convergence times reduces its advantage, the worst case being Timbo(2,2) where the advantage is reduced to little more than 100 generations in 3,000. In the case of F7(2,2), the genetic algorithm still converges an average of 3 times faster than the evolutionary algorithm, despite its 6.2 fold increase in convergence time.

For a mutation radius of 10%, the evolutionary algorithm has a 1.6 fold increase in convergence time for F7(2,2) and a 1.1 fold increase in convergence for Timbo(2,2). The corresponding increases for the genetic algorithm are 0.7 fold (a *decrease*) for F7(2,2) and 1.1 fold for Timbo(2,2). The more diverse search by the genetic algorithm is advantageous with F7(2,2). The 10% mutation radius allows the genetic algorithm to converge on the F7(2,2) minimum at a mean rate faster than zero values can be evolved for F7(0,0).

Note that the genetic algorithm minimized the functions in *all* instances, and with the exception of Rozenbrock, in less than 50,000 generations. In 29 instances of F7(0,0) at 1% mutation radius and in 78 instances of F7(0,0) at 10% mutation, the evolutionary algorithm fails to converge within 50,000 generations. F7(2,2) fails to converge 75 and 28 times, respectively. At a 10% mutation radius, Timbo(0,0) fails to converge twice and Timbo(2,2) 17 times.

5 Conclusion

These results suggest that the more complex the problem hyperspace is, particularly with many dimensions where the likelihood of a torturous search path is high, the greater the benefit of a genetic encoding scheme. While more variable than a traditional evolutionary algorithm, the genetic algorithm can be anticipated to provide an average increase of approximately four fold in rate of convergence in complex hyperspaces.

References

1. Kirchhamer, C. V., Yuh, C-H. and Davidson, H.: Modular cis-regulatory organization of developmentally expressed genes: Two genes transcribed territorially in the sea urchin embryo, and additional examples. Proc. N.A.S. Vo. 93:18 (1996) 9322-9328
2. Lutz, B., Kuratani, S., Cooney, A. J., Wawersik, S., Tsai, S. Y., Eichele, G., and Tsai, M.-J.: Developmental regulation of the orphan receptor COUP-TF II gene in spinal motor neurons. Development Vol. 120 (1997) 25-36
3. De Jong, K. A.: Analysis of the Behaviour of a Class of Genetic Adaptive Systems. PhD Dissertation, Department of Computer Science and Communications Science, University of Michigan, Ann Arbor, M.I., (1975)
4. Davis, L.: Handbook of Genetic Algorithms. Van Nostrand Rienhold, New York (1991)

Generating Lookup Tables Using Evolutionary Algorithms

Tim Hendtlass

Centre for Intelligent Systems

School of Biophysical Sciences and Electrical Engineering,

Swinburne University of Technology, PO Box 218 Hawthorn 3122 AUSTRALIA.

Abstract. This paper describes the use of an evolutionary algorithm to develop lookup tables which consist of an ordered list of regions, each of which encloses training examples of only one category. Compared to a simpler type of lookup table which consists of an unordered list of the training points and their categories, region based tables are smaller and, in general, faster to use. The development of a region based lookup table for the Frey and Slate character recognition problem is described and the size and accuracy achieved are compared with the original Frey and Slate point based lookup table. The reasons why it outperforms the original lookup table are discussed.

1 Introduction.

Lookup tables are a tool for categorizing information. An unknown example is simply assigned the category of the closest member to it in the lookup table. However, lookup tables can be large and consequently finding the closest member may be slow as the distance to all table entries need to be calculated.

This paper considers defining regions of problem space, each of which enclose training examples of only one category. Any member of the test set falling within a region is assigned the category of the training points in that region. This reduces the storage requirements without complicating the lookup process for unknowns that fall within one of the regions. A second special pass is required for unknowns which fall outside any of the defined regions. This finds the region whose boundary is closest to the unknown and infers the category of the unknown from the category of that region. Using an ordered list of regions is efficient: once a point is found to lie in a region no further regions need be checked. The algorithms to categorize an unknown using a point based and an ordered (spherical) region based lookup table are respectively shown in pseudo code in figure one A and figure one B below.

Given a training set of points, no preparatory work is required before the point based algorithm may to be used. Forming the ordered list of regions, each centered at a particular point in problem space and with appropriate dimensions, is, however, non trivial. This is especially true if the categories to be distinguished exhibit any degree of congruency (i.e. the category examples overlap in problem space). The regions cannot be optimized in isolation as the possible set of points for a region to enclose depends on the positions and sizes of the regions earlier in the lookup table. As a

```
Initialize best distance to infinity
Initialize closest example to none
For n = first point to last point
  find distance between unknown and
    point n
  if distance < best distance set best
    distance equal to this distance
  set closest example = example n
Unknown category = category of
  closest example
```

```
Initialize best distance (distance to
  closest region) to infinity
Initialize closest region to none
For n = first region to last region
  x = distance between unknown example
    and center of region n
  y =(x - radius n)
  if x < y set unknown example category
    = category of region n. Exit
  else if x-y < best distance set best
    distance equal to (x-y). Set closest
    region = region n
Finally, if category of unknown example
  not assigned set to category of closest
  region.
```

Fig. 1. Pseudo code for categorizing an unknown example using an unordered point based lookup table (left). Pseudo code for categorizing an unknown example using an ordered region based lookup table.

result the entire set of regions must be developed together. For real data, the combination of all possible orders of all possible regions with all possible positions and sizes is far too large for any type of exhaustive search, but an evolutionary algorithm can permit the development of lookup tables of regions in a realistic time frame.

2 Evolutionary Algorithms.

Evolutionary algorithms use an initial population of (probably) poor solutions to a problem, in this case a number of regions centered on different points in problem space. When breeding new solutions, aspects of (usually two) parent solutions are merged with the aim of producing offspring that combine the best features of, and so out perform, their parents. The offspring are then used as parents to produce another, hopefully better, generation still. The better a solution in a generation performs, the more part it plays in parenting the next generation. In this way a bias towards better performing individuals is introduced. Consequently beneficial traits are assimilated into the population while unproductive traits tend to die out.

As well as combining parts of the parents in the offspring, several types of mutation of the offspring may also occur. This has the effect of introducing new material into the gene pool. For each possible type of mutation, the probability that a mutation will occur must be specified and, if appropriate, so must the maximum magnitude that such a mutation may take.

Evolutionary algorithms are computationally expensive as solutions tend to be slow to evolve. In general, the larger the population size, the richer the pool of genetic material from which to construct offspring, the lower the number of generations to a suitable solution but the more processing is required for each generation. As a result, the computational expense of the solution is high even for large populations. The smaller the population size, the more important a role mutation plays in providing diversity in the genetic material from which the new solutions are built. Ever increasing the mutation, however, will show a diminishing return: excessive mutation is essentially equivalent to reinitializing the population each generation.

Evolutionary algorithms are very efficient at performing a broad search of a vast solution space but they are not, in general, able to find the optimum solution in a reasonable time. To more rapidly reach the optimum solution the evolutionary algorithm may be teamed with a local search algorithm (see, for example, [1]) which explores the regions of problem space around the points of interest found by the evolutionary algorithm.

2.1 Details of the evolutionary algorithm used.

The chromosome used to describe a particular lookup table consists of a fixed number of genes, each of which describes one region of problem space. The description consists of the center coordinates of the region, the radius of the region and the category to be assigned to any example which falls within that region. The order of the genes, from first to last, describe the order of the regions in the lookup table.

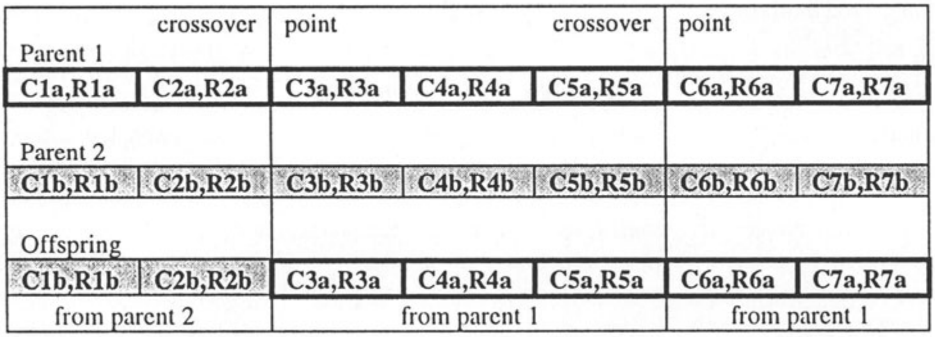
A new population of lookup tables is bred from the previous population of lookup tables as follows:

- The *n* best performing lookup tables (the elite) are copied directly into the next generation without any modification.
- The *n* worst performing lookup tables are discarded and replaced by copies of the elite mutated as described below.
- The remaining places in the population are filled with new lookup tables derived from the existing population using both cross over and mutation.

The breeding probability of each member of the existing population is derived as follows: Let *P(i)* be the number of lookup tables with poorer performance than the i^{th} lookup table on every criterion (the criteria used are: the number of training points correctly enclosed by the regions of the lookup table, the number of points wrongly enclosed and the number of non-empty regions). Let *S* be the total number of solutions. The breeding probability of the i^{th} lookup table is defined by the Pareto Optimality [2] to be:

$$B_i = (1 + P(i)) / (\sum_{n=1}^{n=S} (1 + P(n)).$$

A conventional crossover mechanism is used which involves two parents. The probability that a particular lookup table be chosen as one of the parents is proportional to that lookup table's breeding probability. Once the two parents are selected, a predefined number of crossover points are chosen along the length of the chromosome and the child inherits the regions between a pair of points from one parent. At a crossover point a new choice is made for the parent who will contribute the regions up to the next crossover point. The child lookup table inherits complete



Parent 1	crossover	point		crossover	point	
C1a,R1a	C2a,R2a	C3a,R3a	C4a,R4a	C5a,R5a	C6a,R6a	C7a,R7a
Parent 2						
C1b,R1b	C2b,R2b	C3b,R3b	C4b,R4b	C5b,R5b	C6b,R6b	C7b,R7b
Offspring						
C1b,R1b	C2b,R2b	C3a,R3a	C4a,R4a	C5a,R5a	C6a,R6a	C7a,R7a
from parent 2		from parent 1			from parent 1	

Fig 2. An example of the crossover mechanism used.

regions from the parents, so no region in the child is defined by some region parameters from one parent and other parameters from the second parent. An example of the crossover mechanism is shown in figure two above.

Two types of mutation are used: firstly, a random mutation of the region center position and, secondly, moving a region containing either no or only one example so that it is centered on an example from the training set that is not enclosed by any region in this solution. The first mutation type is quite standard and permits regions to move, hopefully to a more advantageous position, while still describing the same group of like examples. The second permits regions to be developed in previously unexplored regions of problem space. Moving a region that contains one example may seem counterproductive as the local search algorithm described below ensures that each region grows to the largest possible size such that it only contains one category of example. As a result it may be felt that a region with only one enclosure does describe a singular example amongst a number of examples of different type. However, single example region moving can help the performance of a lookup table by filtering perplexing examples from the training set (see section 3).

2.2 Details of the local heuristic used.

Once the position of a region in problem space has been determined by a combination of crossover and mutation, a local search algorithm is used to set the optimum radius. This increases the radius of the region from zero, noting the first unclaimed point enclosed by the region (an unclaimed point is one that has not been enclosed by any other region earlier in the lookup table). The region is assigned the category of the first enclosed point. The region radius is increased until the first unclaimed example of another category is found. The radius is set so that the region boundary is set half way between the last example of the region's category and the first example of another category. This algorithm, shown in figure three below, together with ordering the regions from the smallest to the largest, helps prevent having small regions existing totally within larger regions of the same category.

2.3 The data set.

The character recognition data set introduced by Frey and Slate [3] is a well studied problem which consists of 16,000 examples of the uppercase letters of the English language from 20 different fonts to which varying amounts of distortion have been applied. A further 4,000 examples are used as a test set. This data set is known to exhibit a significant degree of congruency between the categories and may be considered typical of complex real life categorization problems. Each example is described by sixteen numbers together with a category. Using the 16,000 training examples as a conventional lookup table, 3,827 of the 4,000 test examples (95.675%) are correctly identified.

A number of other attempts have been made to produced a classification system for this data which requires less memory than that used by the conventional lookup table. These have included single or multiple artificial neural nets and evolved polynomial classifiers. The performance has varied from approximately 67% to about 93.6% correct on the test set. ([For a list of results up to 1994, see [4]. One, as yet unreported, technique uses a set of 26 recognizers, each of which attempts to identify one letter using an ensemble consisting of artificial neural nets and polynomial data partitioners. It has achieved 96.2% correct, the improvement over the lookup table being due to the better generalizing capabilities of the ensembles. [5].

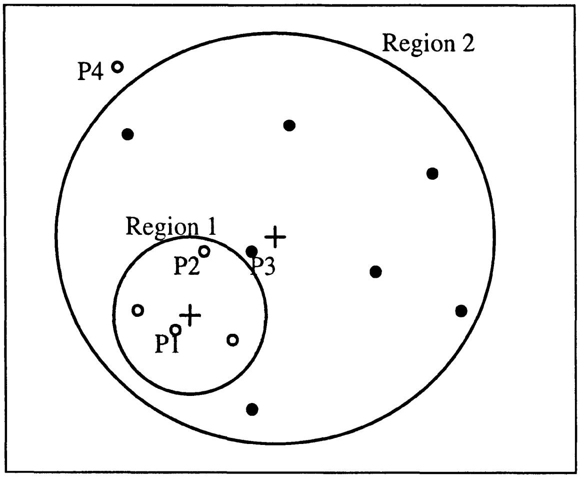


Fig 3. Region 1has it's category defined by the point P1 which is closest to the origin of region 1 (the cross). Region 1 expands to between points P2 and P3, P3 being the closest point of another category and P2 the last point enclosed by region 1 before P3 is reached. P3, as the closest point to the origin of region 2, defines the category of region 2. Note: P4 cannot exist in region 1 with the current region origins. Organizing the table in ascending order of region radius permits small regions inside larger ones: region 2 can claim the whole area it covers as any test point falling inside region 1 will never be seen by region 2.

3. Results And Discussion.

The best performing region based lookup table generated correctly classified 3,961 of the 4,000 testing examples, a performance which exceeds that of the conventional lookup table on the same data while requiring only just over 13% of the storage space. To be fair, the conventional lookup table had no ability to infer a result. The best performing region based lookup table only got 85.6% correct by enclosure, the rest being by inference. This suggests that the test set contained a number of outliers compared to the character groupings in the training set.

Lookup tables were ranked for the purpose of deciding which was the best table using the following formula:

$$\text{Score} = 10{*}\text{NCE} - 2{*}\ \text{NWE} - \text{NR}$$

where NCE is the number of examples enclosed in a region of the same type, NWE is the number of examples enclosed in a region of another type, and NR is the number of non-empty regions. Fuller experiment details and results are shown in table one below.

Experiments with other crossover and mutation control values indicate that performance better than that of the simple lookup table can be achieved over a range of control values.

The better final performance achieved by the region based table suggests that it has acquired some generalization capability. This is a consequence of the examples misclassified during training, together with the swapping of regions containing singular points. Misclassification can arise as follows. A new lookup table's regions are first ordered from smallest to largest based on the radii of the regions in their respective parents. Then the radii are readjusted in order, which may result in dramatic changes in the radii of several regions. The further size based resorting before scoring may therefore significantly change the order of the regions in the solution so that, after final scoring, the examples in some regions may now be different from those enclosed by the same regions during the radius adjusting process.

	Conventional Lookup Table	Best Region Based Table
Training Result Correct	100%	15,470 (96.69%)
Training Result Wrong	not applicable	114 (0.71%)
Training Result Unclassified.	not applicable	416 (2.6%)
Testing Results correct by enclosure	not applicable	3,424 (85.6%)
Testing Results correct by inference	not applicable	537 (13.4%)
Testing Results total correct	3827 (95.68%)	3,961 (99%)
Relative storage requirements	1	0.1322

Tablel. Best region based lookup table results. *Experimental parameters:* Population of 100 lookup tables. One point crossover. 25% probability of each type of mutation. It took 342 generations to breed the table above which consisted of 1,991 regions.

In particular, some examples may now appear in a region of another category. When the region was sized, these examples had been already claimed by another region proceeding it in the table. Now, however, that previously claiming region either no longer proceeds it or is no longer large enough to enclose the example in question.

A significant number of the examples misclassified in this way, as well as those singular points unclassified as a result of singular example region swapping, are perplexing examples. Perplexing examples are examples which are inconsistent with the majority of examples and whose inclusion in the training set degrades recognition accuracy. These processes remove some perplexing examples (along with non perplexing ones) and thus improves the training performance. Without doubt, the testing set also contains perplexing examples which no recognition system will be able to correctly identify. The best possible lookup table, constructed by ignoring the perplexing examples, will give the best recognition of the non perplexing examples in the test set.

4. Conclusion.

A region based lookup table has comparable performance to a conventional lookup table, while requiring less storage space and being faster to use. For data containing perplexing examples, the performance of the region based lookup table can exceed that of the conventional table. The price paid for these advantages is the extra processing required to develop the list of regions. However, since this need only be done once for a given data set, it is a minor price to pay.

References

[1] John R Podlena and Tim Hendtlass: Using a combination of Genetic Algorithm and Hill Climbing Techniques for the solution of a difficult letter recognition problem. Proc. Of the Eighth IEA/AEI International Conference, 1995 (55-60)

[2] Carlos Fonseca and Peter Flemming: An overview of Evolutionary Algorithms in Multiobjective Optimisation. Evolutionary Computing, volume 3, number 1. 1995. (1-16)

[3] P.W.Frey and D.J.Slate: Letter recognition using Holland Style Adaptive Classifiers. Machine Learning, volume 6 number 2, (1991).

[4] G Jones: Letter Image Recognition Data www.sgi.com/Technology/mlc/db /letter.names

[5] D. Philpot, personal communication.

A Genetic Algorithm for Linear Feature Extraction

Suchendra M. Bhandarkar, Jögg Zeppen, and Walter D. Potter

Department of Computer Science, The University of Georgia
Athens, Georgia 30602-7404, U.S.A.

Abstract. A genetic algorithm for the detection and extraction of linear features in a gray scale image is presented. Conventional techniques for detection of linear features based on template matching and the Hough Transform, rely on an exhaustive search of the solution space, thus rendering them computationally intensive, whereas techniques based on heuristic search in a state-space graph are prone to being trapped in a suboptimal solution state. On account of its *building blocks* property the genetic algorithm alleviates the need for exhaustive search and the stochastic nature of the genetic algorithm operators makes it robust to the presence of local optima in the solution space. Experimental results on gray scale images bring out the advantages of the genetic algorithm in comparison to the template matching-based and Hough Transform-based techniques for linear feature extraction.

1 Introduction

Detection and extraction of linear features in a gray scale image has been a thoroughly researched topic in computer vision. The feature detection and extraction phase typically follows edge detection. During edge detection, pixels in the gray scale image that denote significant changes in image intensity are marked as *edge pixels*. The feature detection and extraction phase subsequently groups the edge pixels into *features* where these features are expected to denote contours or boundaries of objects of interest in the image. In this paper we consider the detection and extraction of linear features i.e., features that can be represented or approximated by a piecewise combination of straight line segments.

Traditional techniques for detection of linear features in an image are based on (a) template matching [9], (b) the Hough Transform [4] or (c) heuristic search in a state-space graph [1]. In template matching schemes, a set of templates for the feature of interest is defined. Each template is then moved systematically across the edge image. At each position of the template, a correlation coefficient or match metric between the template and the underlying portion of the image is computed. A high value for the correlation coefficient or match metric denotes the existence of the corresponding feature at that position. Template matching techniques entail an exhaustive search of the template space and image space. If a set of k templates of size $M \times M$ are used on an image of size $N \times N$ then the computational complexity of the template matching procedure is $O(kM^2N^2)$.

In the case of the Hough Transform, the linear feature described by the parametric equation $r = x\cos\theta + y\sin\theta$ is represented by a point in the (r, θ) parameter space where r is the perpendicular distance of the linear feature from the origin and θ is the angle made by the perpendicular with the x axis. The parameter space is discretized in the form of a two-dimensional array or accumulator consisting of a finite number of bins. Each edge pixel casts a *vote* for a particular bin if it satisfies the parametric equation of the linear feature with the specific parameter values that the bin represents. The maxima in the accumulator bins hypothesize the presence of linear features with the corresponding parametric representation(s) in the image. Maxima detection is followed by contour tracing in order to verify the existence of the feature with the corresponding parametric representation in the image.

The asymptotic computational complexity of the Hough Transform is $O(kN^2)$ where k is the number of levels in the range $[-\frac{\pi}{2}, \pi]$ into which the angle θ is quantized and $N \times N$ is the image size. The Hough Transform also has a space complexity of $O(kN)$ resulting from the size of the accumulator array. The Hough Transform can be optimized for speed and memory by exploiting the intensity gradient direction information [2]. This results in a Hough Transform with time complexity of $O(N^2)$ and a space complexity of $O(N)$. Since the computation of $\cos\theta$ and $\sin\theta$ is typically time-consuming most implementations of the Hough Transform precompute the values of $\cos\theta$ and $\sin\theta$ and store them in a table. With table look-up, the run time of the Hough Transform is improved but the asymptotic computational complexity is still $O(kN^2)$ without gradient information and $O(N^2)$ with gradient information. The space complexity of the Hough Transform with table look-up is $O(k + N^2)$ without gradient information and $O(k + N)$ with gradient information.

In heuristic search-based techniques, the edge pixels are represented by nodes in a graph. Each node in the graph is assigned a weight in proportion to the magnitude of the image intensity gradient (i.e., edge contrast) at the corresponding edge pixel. An arc between two adjacent nodes (i.e., adjacent pixels) is assigned a weight in inverse proportion to the difference in the gradient orientations at the two nodes. A suitable cost function is defined which takes into account the node and arc weights. An optimal feature is then represented by a path in the weighted graph that minimizes the cost function. Various heuristic search strategies such as A* search and dynamic programming could be used for determining this optimal path. Heuristic search techniques, have a tendency to get trapped in a locally optimal solution whereas determining the globally optimal solution results in high combinatorial complexity.

The Genetic Algorithm (GA) [5,6] is a population-based stochastic optimization procedure that mimics the natural process of biological evolution. By exploiting the diversity within a population of solutions, the GA alleviates the undesirable but inevitable tradeoff between the need for an globally optimal (or near-optimal) solution and the high complexity of exhaustive search that guarantees an optimal solution. Central to the GA are the concepts of *population*, *fitness*, *chromosomes*, *selection*, *crossover* and *mutation*. Potential solutions to a combinatorial optimization problem are represented as bit strings or *chromosomes.* A collection of potential solutions or chromosomes constitutes a *population.* With each chromosome is attached a *fitness* value which is a measure of goodness of the corresponding solution. The fitness value is computed using a *fitness function* which is derived from the objective function to be optimized and the constraints underlying the combinatorial optimization problem.

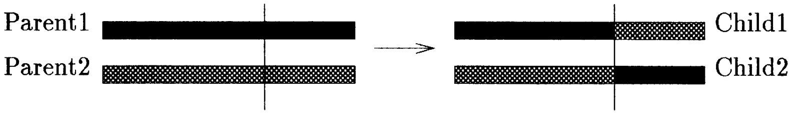


Fig. 1. The crossover operation

The chromosomes from a given population are chosen using a *selection* operator to form a mating pool for reproduction. The *roulette wheel* selection operator, which selects each chromosome with a probability proportional to the ratio of the fitness of the chromosome to the overall fitness of the population, is a popular choice. Two mates, selected at random from the mating pool, reproduce via the *crossover* operator. During crossover, a point along the length of the chromosome is selected at random and the ends of the chromosomes swapped with a predefined crossover probability to generate a pair of offspring for the next generation Figure 1. Each of the offspring then is subject to a random localized change via the *mutation* operator, which in our case amounts to flipping each bit of the offspring with a predefined mutation probability. The above operations are carried out repeatedly until a new population of offspring is generated which then replaces the current population. The GA typically iterates through several generations using the selection, crossover and mutation operators to replace the current population with the succeeding generation. The GA is considered to have converged if the fitness value of the best member in the population has not changed substantially over the past few successive generations.

The GA has several advantages over conventional search procedures. The selection and crossover operators enables useful subsolutions, referred to as *building blocks* or *schema* in the GA literature, to be propagated and combined to construct better and more global solutions with every succeeding generation [6]. The mutation operator, which represents a random local search, enables the GA to escape from local optima in the search space. The population of candidate solutions and the GA operators enables one to explore a diversity of solutions and hence a larger fraction of the search space. The *Schema Theorem* [5,6] enables the GA to sample a large fraction of the search space even with a relatively small population size. This increases the chances of the GA being able to arrive at a globally optimal solution.

2 GA for Linear Feature Extraction

Linear feature extraction using the GA consists of two phases; the edge detection phase, and the feature extraction phase where the GA is actually used.

2.1 Edge Detection

The edge pixels in the image are detected using an edge detector. In our case, we used the Laplacian-of-Gaussian (LoG) edge detector [7] followed by a procedure to detect the zero crossings in the image resulting from convolution with the LoG operator. The 2-D LoG operator is given by

$$\nabla^2 G(x,y,\sigma) = \left[\frac{1}{\pi\sigma^4}\left(\frac{x^2+y^2}{2\sigma^2} - 1\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right] \tag{1}$$

The image resulting from the convolution of the LoG operator with the image $F(x,y)$ is given by

$$F_g(x,y,\sigma) = \nabla^2 G(x,y,\sigma) \star F(x,y) \tag{2}$$

where $\star$ denotes the convolution operation. The zero crossings in the image $F_g(x,y,\sigma)$ are determined by convolving it with a set of masks $M_1 = [-1,1]$, $M_2 = [-1,1]^T$, $M_3 = [-1,0,1]$ and $M_4 = [-1,0,1]^T$ such that

$$F_e(x,y,\sigma) = \begin{cases} 1 \text{ if } \sum_{i=1}^{4} |M_i \star F_g| > \tau \\ 0 \text{ otherwise} \end{cases} \tag{3}$$

where $F_e(x,y,\sigma)$ is the edge map at scale σ and τ is a predefined threshold. In our case, we chose $\sigma = 3$.

2.2 Linear Feature Extraction

Since most edge detectors have very local support, they are vulnerable to noise and missing data. The GA-based linear feature extraction procedure described here is capable of rejecting noisy edge pixels and is also robust to missing edge pixels. The outline of the GA-based linear feature extraction procedure is given in Figure 2.

For each new starting point (x,y), a population of chromosomes is created such that each chromosome represents a line with its starting point in the neighborhood of (x,y) and at a randomly chosen orientation in the range $[0^o, 180^o]$. Each chromosome is represented by 25 bits where the interpretation of the various bits is as follows:

Bit Positions	Number of bits	Representation
0–3	4	Δx
4–7	4	Δy
8–15	8	θ
16–25	9	l

```
mark all edge pixels as "unvisited";
repeat
 {
   raster scan the image until an "unvisited" edge pixel is encountered;
   make this pixel the staring point of a new line;
   determine the best line using the GA;
   mark all the pixels on this line as "visited";
 } until all the edge pixels are marked as "visited";
```

Fig. 2. Outline of the linear feature extraction procedure

Here Δx and Δy are the x and y deviations about the starting points of the line, θ is the orientation angle of the line and l is the length of the line. Assigning nonzero values to Δx and Δy enable us to vary the starting position of the line in a small neighborhood about the chosen point (x, y). In our case, $-2 \leq \Delta x \leq 2$ and $-1 \leq \Delta y \leq 3$.

The fitness of a chromosome is computed as follows:

$$f = w_1(n_h - n_m)^\alpha + w_2 \left(\frac{1}{1 + \sum \epsilon^2} \right)^\beta \tag{4}$$

Here, n_h is the number of pixels on the line that have a corresponding pixel in the edge image F_e whereas n_m is the number of pixels on the line that do not, $\sum \epsilon^2$ is the sum of squared deviations of the edge pixels in F_e in a predefined neighborhood of the line, α and β are predefined exponents and w_1 and w_2 are predefined weights. The neighborhood of a line is defined by two lines parallel to the given line at a distance of δ on either side. In our case we chose $w_1 = 1$, $w_2 = 0.6$, $\alpha = 2$, $\beta = 1$ and $\delta = 2$. The line corresponding to a chromosome and the parallel lines defining its δ neighborhood were generated using the midpoint line drawing algorithm [10] and overlaid on the edge image. The lines were drawn from their starting points until they intersected the image border.

Once a line was accepted, the corresponding pixels in the edge image were marked as *visited* and were removed from consideration as starting points for other lines. However, visited pixels were still considered for the purpose of computation of the fitness values for the remaining lines to account for the fact that lines may intersect each other.

One of the problems with the standard crossover operator (also referred to as a *single-point* crossover) is that it tends to disrupt good chromosomes [8]. An alternative is to use *multi-point* crossover with a crossover point within each encoding region of the chromosome. In our case, a crossover point was chosen within each of the four encoding regions of the chromosome corresponding to the parameters l, θ, Δx and Δy. A four-point crossover operator which does not violate the boundaries of the encoding regions was implemented (Figure 3).

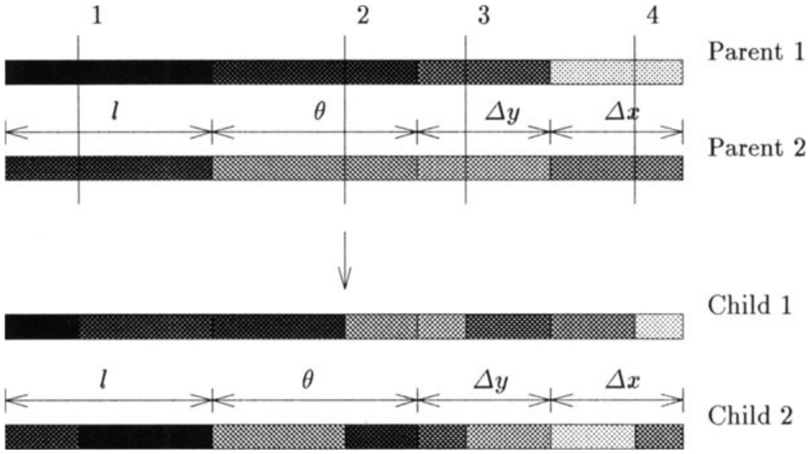


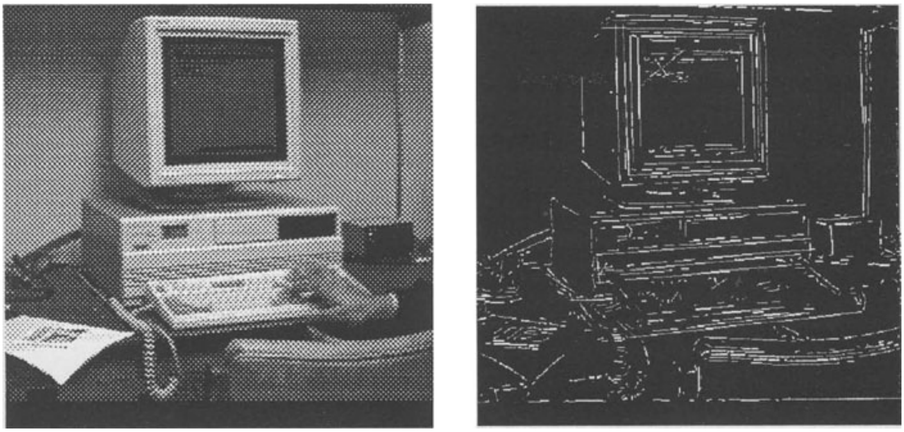
Fig. 3. The four–point crossover

3 Experimental Results

The GA-based linear feature extraction technique was implemented and tested on a set of gray scale images. The GA-based linear feature extraction technique was implemented with both, single-point crossover (Figure 1) and four-point crossover (Figure 3). In our implementation of the GA, we chose a population size of 50 in each generation and the probability of crossover $p_c = 0.6$ and a probability of mutation $p_m = 0.07$. The fact that the fitness value of the best chromosome in the population had not changed over the past five consecutive generations was used as the halting criterion or convergence criterion. Based on this criterion, the GA was found to converge within 25 generations with single-point crossover and within 16 generations with 4-point crossover in all our experiments.

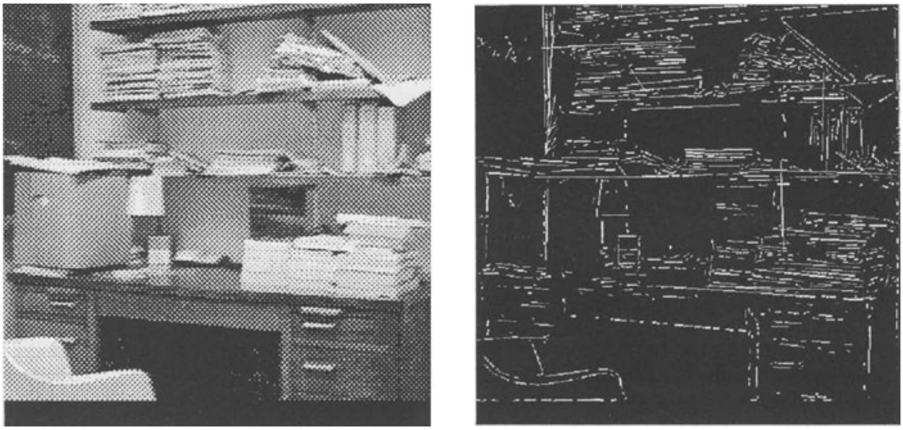
Three of the images that were used in our experiments are shown in Figures 4(a), 5(a) and 6(a) and are referred to as *Computer*, *Desk* and *Bookshelf* respectively. All the gray scale images were 512 × 512 pixels in size with a gray scale resolution of 8 bits per pixel (i.e., 256 gray levels). Figures 4(b), 5(b) and 6(b) show the linear features for the corresponding gray scale images. The linear features were extracted using the GA with four-point crossover.

For the sake of comparison, the GA-based linear feature extraction technique, the Hough Transform and the template matching algorithm were implemented on a 110 MHz SUN SPARCStation 5 with 64MB RAM. The template matching technique used was a variation of the one proposed by Nevatia and Babu [9]. A set of 12 templates of size 5 × 5 were used to detect linear features at orientations in steps of 30^o in the range $[0^o, 360^o]$. A finer resolution in orientation angle



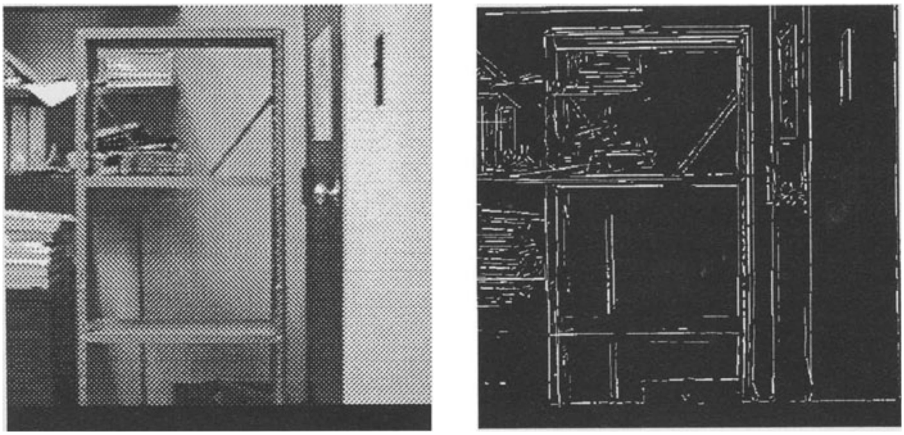
(a) (b)

Fig. 4. (a) Gray scale *Computer* image (b) Linear features detected using the GA



(a) (b)

Fig. 5. (a) Gray scale *Desk* image (b) Linear features detected using the GA



(a) (b)

Fig. 6. (a) Gray scale *Bookshelf* image (b) Linear features detected using the GA

could have been obtained but at the cost of a greater number of templates, with each template of larger size, which would have consequently entailed a higher execution time. The response of each edge pixel to the set of templates was computed and the template with the highest match at each pixel location was noted. Adjacent edge pixels with similar template labels were grouped into linear features with the corresponding template orientation. Short linear features or isolated edge pixels were rejected as noise. Two implementations of the Hough Transform were attempted; one that did not use intensity gradient information and one that did. The non-gradient-based Hough Transform implementation used a 2-D (r, θ) accumulator of size 724×270 where r was quantized to a resolution of 1 pixel and θ was quantized to a resolution of 1^o. The gradient-based Hough Transform implementation used a 1-D r accumulator of size 724. The intensity gradient direction value θ was computed using a smoothing gradient operator [3]. The value of r was computed for different values of θ' in the range $[\theta - \Delta\theta, \theta + \Delta\theta]$ where $\Delta\theta$ was a suitably chosen threshold to account for error in the intensity gradient computation due to noise in the image. In our case we chose $\Delta\theta = 5^o$ with θ' incremented in steps of 1^o. The values of $\cos\theta$ and $\sin\theta$ were precomputed and stored in a table which was then looked up during the Hough Transform computation.

Table 1 compares the execution time of the GA with the execution times of the Hough Transform and the template matching algorithm for each of the three aforementioned images. As can be seen from the Table 1, the Hough Transform was computationally the most intensive whereas the template matching algorithm was the fastest. It is to be noted that the template matching algorithm gave an orientation resolution of only 30^o compared to 1^o in the case of the Hough Transform and 0.7^o in the case of the GA. A template matching algorithm with

Table 1. Execution time in seconds of the GA (with one-point and four-point crossover), the Hough Transform (with and without intensity gradient computation) and the template matching algorithm on a SUN SPARCStation 5 (110 MHz, 64MB RAM) for each of the test images: *Computer*, *Desk* and *Bookshelf*

Image	GA		Hough Transform		Template Matching
	(1-point)	(4-point)	(non-gradient)	(gradient)	
Computer	312	295	772	603	288
Desk	328	309	723	582	302
Bookshelf	307	281	661	512	273

an orientation resolution of 1^o would have been computationally far more intensive. The Hough Transform is computationally intensive since it performs an exhaustive search. Exploiting the intensity gradient direction information improves the overall performance of the Hough Transform. However, the reduction in the dimensionality of the Hough accumulator from $O(N^2)$ to $O(N)$ and the reduction in the complexity of the vote accumulation procedure from $O(kN)$ to $O(N)$ is partly offset by the computation of the intensity gradient direction at each pixel location. The GA on the other hand, avoids exhaustive search by propagating useful subsolutions (or schema) from one generation to the next and by combining useful subsolutions to yield more global solutions with every succeeding generation. The stochastic nature of the selection, crossover and mutation operators also makes the GA robust to the presence of local optima in the search space. The four-point crossover results in faster convergence of the GA in terms of number of generations; from an average of 25 generations for single-point crossover to an average of 16 generations for four-point crossover. However, the fewer number of generations needed for convergence in the case of the GA with four-point crossover is partially offset by the fact that the implementation of the four-point crossover is computationally more complex than the single-point crossover. The execution times of the GA as shown in Table 1 and the resulting images (Figures 4(b), 5(b) and 6(b)) indicate that the GA is capable of generating high-quality (i.e., near-optimal) solutions without resorting to exhaustive search.

4 Conclusions and Future Directions

In this paper we presented a GA-based feature extraction algorithm that is capable of extracting linear features in an image. The GA was compared to the Hough Transform and the template matching algorithm for the extraction of linear features in a gray scale image. The Hough Transform and the template matching algorithm perform an exhaustive search and were hence seen to present an undesirable but inevitable tradeoff between the accuracy of feature extraction/localization and the resulting computational complexity. Reduction in computational complexity entails reduction in the spatial resolution of the image and/or reduction in the resolution of the feature parameters, which in this case, are the position and orientation of the linear feature. Reduction in the spatial resolution of the image and/or the resolution of the feature parameters results in reduced accuracy of the extracted features. Conversely, increasing the accuracy of the extracted features implies greater computational complexity. The GA is able to alleviate the tradeoff between the accuracy of feature extraction/localization and the computational complexity by providing high-quality (i.e. near-optimal) solutions without the need for exhaustive search with its resulting high computational complexity.

With regard to future research, we plan to investigate the effect of different chromosome encoding schemes on the performance of the GA in the context of feature extraction. The manner in which the line parameters (starting position, orientation and length) were encoded in this paper was *ad hoc.* We intend to study the optimal chromosomal encoding of image features for efficient and accurate feature extraction.

References

1. Ashkar, G.P., and Modestino J.W.: The contour extraction problem with biomedical applications. *Comp. Graph. Img. Proc.* 7, (1978) 331–355.
2. Ballard, D.H.: Generalizing the Hough Transform to detect arbitrary shapes. *Pattern Recognition.* 13(12) (1981) 111-122.
3. Canny, J.F.: A computational approach to edge detection, *IEEE Transactions on Pattern Analysis and Machine Intelligence.* 8(6) (1986) 679–698.
4. Duda, R.O., and Hart, P.E.: Use of the Hough transformation to detect lines and curves in pictures. *Comm. ACM.* 15(1) (1972) 11-15.
5. Goldberg, D.: *Genetic Algorithms in Search, Optimization and Machine Learning.* Addison-Wesley Pub. Company, Reading, MA (1988).
6. Holland, J.H.: *Adaptation in Natural and Artificial Systems.* University of Michigan Press, Ann Arbor, MI (1975).
7. Marr, D., and Hildreth, E.: Theory of edge detection. *Proc. Royal Soc. London.* B 207 (1980) 187–217.
8. Mitchell, M.: *An Introduction to Genetic Algorithms.* MIT Press, Cambridge, MA (1996).
9. Nevatia, R., and Babu, K.R.: Linear feature extraction and description. *Comp. Graph. Imag. Proc.* 13(3) (1980) 257–269.
10. Sproull, R.F.: Using program transformations to derive line drawing algorithms. *ACM Trans. Graphics.* 1(4) (1982) 259–273.

Knowledge Representation in a Blackboard System for Sensor Data Interpretation

S M C Peers
Department of Mechanical Engineering, University College London,
Torrington Place, London WC1E 7JE, United Kingdom

Abstract. A prototype system has been developed for automated defect classification and characterisation of automotive or other components employing two separate inspection sensors, vision and electromagnetic. The development work for the electromagnetic sensor sub-system was fraught with difficulties. In particular, there was the basic problem of encoding human expertise. The reasoning carried out by the human inspectors is more complex than the experts themselves may suppose and was not easily encapsulated.

A blackboard architecture was used to integrate the different areas of expertise required for each sensor to interpret the results of the inspections. One issue here discussed is the effective use of the blackboard architecture for intelligent data fusion at all levels to improve interpretation.

1. Introduction

The primary requirements of an automated inspection system are to identify defects, surface condition and quality. For this, it is best to use two or more non-destructive testing methods based on different physical principles, such as close visual inspection and the ACFM technique. To automate the inspection process requires methods of interpreting the sensor data. It was found that it is not a trivial exercise to identify these methods fully. Formal interpretation techniques may exist but may be based on idealised conditions and how they are employed by experts is often modified, perhaps subconsciously.

In a past project, a prototype AIRES (Automated Image Reconstruction using Expert Systems) system was developed which combined vision and electromagnetic sensors to detect and characterise surface flaws in automobile components. The sensor data was interpreted by especially developed software modules, integrated into a blackboard system. The classification and characterisation routines used classical signal processing algorithms and neural networks. This paper describes the software architecture used to integrate the different areas of expertise necessary to provide automated defect classification and characterisation using electromagnetic sensors and some of the issues. Current proposals consider extending the ACFM sensor work presented here, in particular with respect to neural networks and data fusion.

2. The AIRES System

The AIRES project used electromagnetic and vision inspection techniques to detect and characterise defects in machined metal components. Defects are first identified, then classified and characterised automatically by making use of expert knowledge about the inspection sensor employed and then by combining the results of both. One way of automating the interpretation of inspection data is to produce software that imitates the procedures and reasoning used by a human expert inspector. For this the software will need to store and manipulate information related to the component being inspected, the sensor, procedures for sensor data interpretation, and a comparison of different inspection techniques.

2.1 The Architecture of the Knowledge Based System

The electromagnetic and vision sensor data analysis routines were incorporated within a knowledge based system (KBS). The KBS uses a blackboard architecture, considered the most general and flexible architecture [1]. The sub-systems of the KBS (see Fig. 1) are the electromagnetic module including neural networks (Electromagnetic Sensor KBS), the vision module and neural networks (Vision Sensor KBS), the Data Fusion and the Graphical User Interface.

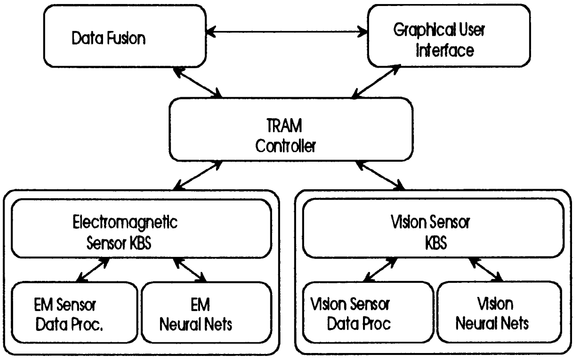


Fig. 1 Knowledge based system structure for AIRES

2.2 The Vision Sensor

The vision sensor consisted of a monochrome CCD camera, with computer controlled focus and lighting. Knowledge sources were developed for dedicated image processing hardware based on standard methods [2] for detection of surface defects, with intelligent control to select the best available method for a particular situation. These were augmented by the use of vision neural nets.

2.3 The Electromagnetic Sensor

The electromagnetic inspection technique applied here is based on the Alternating Current Field Measurement (ACFM) method. This method works by inducing a high frequency alternating current in the component inspection surface and studying the resulting magnetic field (B) above the surface. The data is collected as three matrices of B values, one matrix for each direction and each point in the matrix for a sample location. The 3-D surface of the magnetic field can be analysed for signs of defects, for their type (pit, crack, or other) and their parameters. The algorithms used were based on the theoretical modelling of ACFM. Neural net routines were also used as a complement to the ACFM theoretical algorithms.

2.4 Data Fusion

Combining the results of two sensors will give a more robust and powerful system. In this case, the combination of visual data and the electromagnetic data can work well. Both the electromagnetic and vision techniques can detect surface defects only. The electromagnetic technique works through coatings and can, given good conditions and particular defect geometries, give accurate defect sizes, in particular for cracks. The visual technique is somewhat more robust in terms of required conditions for initial detection of defects, but cannot give depth measurements, although it gives better results for pits. Making use of the vision data early on would provide more accurate initial interpretations of the electromagnetic data. Consequently, data fusion merges the interpreted results from the electromagnetic signals and the vision data.

2.5 Applications

This multi-sensor system is for inspection of metallic components such as are produced in the automotive and aerospace industries. The approach is intended to be applicable to on-line defect detection during production. The requirements for the AIRES system was that it should be able to inspect standard plate specimens with features such as pits, cracks, slots etc., and, at a later stage, a camshaft with surface defects on ground faces.

3. Interpretation of the Electromagnetic Sensor Data

Within AIRES, the electromagnetic sensor data needs to be processed and the results interpreted. The electromagnetic sensor used in this work was based on the *alternating current field measurement (ACFM)* technique [3]. This technique was developed for the detection and sizing of surface cracks and defects in metallic structures. Under appropriate conditions, if an alternating current is induced in a surface, the magnetic field above the surface will vary according to the geometry of the surface. Thus the ACFM sensor induces a current in the surface, by an inducing coil suspended over it, and the magnetic field B above the surface is recorded.

3.1 The Electromagnetic field Data

The precise numerical form of the magnetic field above a crack can be accurately predicted. By comparing the experimentally recorded shape to the predicted shape, the crack dimensions may be established [4]. A similar method may be used to detect and size any other defect, such as a pit, for which the numerical magnetic field can be predicted. The practical use of the numerical solutions requires these to be implemented as look-up tables. The use of these tables requires a defect to be identified and its alignment given. An example of the magnetic field components

B predicted by electromagnetic theory for a crack, length 8mm and depth 1mm in a flat surface, is shown in Fig. 2.

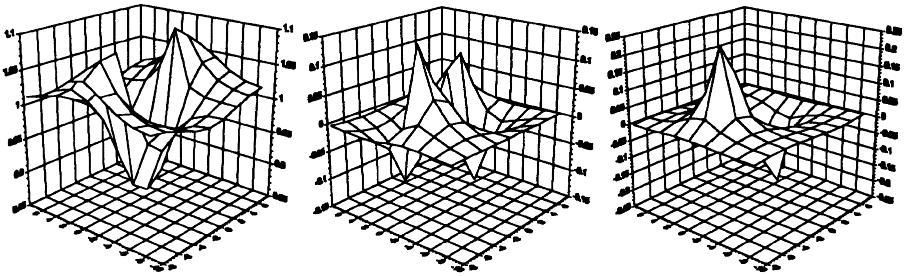


Fig. 2. The magnetic field B (B*x*, B*y* and B*z*) over a crack

3.2 The ACFM Sensor

The sensor used in AIRES to record the magnetic field data is made of an inducing solenoid suspended above an array of magnetic field sensors, which are cylindrical coils. A coil can only record the field in the direction of its axis, hence they are placed alternating in each of the three *x*, *y* and *z* directions. To record a complete set of magnetic field data, the sensor is placed in three adjacent positions on the surface. The induced field is not uniform, so to apply a first-order correction to this non-linear phenomenon, the probe is used to scan a flat featureless surface and this background reading subtracted from scans. The ACFM sensor outputs matrices for the *x*, *y*, and *z* components of the magnetic field B and these matrices are interpreted to find the geometry and size of the defect.

3.3 Typical Theoretical electromagnetic Data

From Fig. 2, where a pattern in the B fields for detecting can be seen, a human expert is able to classify or at least detect a defect. The patterns can be described in terms of peaks and troughs. The initial work concentrated on classification based on attempting to encapsulate the reasoning that an expert would carry out for classification. The important characteristics of the B surfaces are:

Bx **A trough lying in between two peaks.** The length of a crack is related to the distance between the B*x* peaks from electromagnetic theory. The depth of a crack may be also be calculated in a similar way from the peak and trough co-ordinates and values. This is also true for other defect geometries.

By **Two peaks and two troughs.** The relative positions of these are used to distinguish between pits and cracks.

Bz **One peak and one trough.** The Bz peak and trough co-ordinates and values are used as supplementary data.

The procedure for interpretation of the magnetic field from the ACFM requires finding the co-ordinates and the values of the peaks and troughs for each of the B*x*, B*y* and B*z* data sets. An identified feature is classified as either a pit, a crack or of unknown type, by using the four B*y* turning points (see Fig. 3). If the angle α between the lines joining opposite pairs of them is small, the defect is a crack; if α is large, the defect is a pit. The value of α outside given ranges will signify a unknown defect type. If a crack or pit exists, the length and depth are then calculated by using the co-ordinates and values of the B*x* and B*z* peaks and troughs and a set of look-up tables for cracks or for pits which relate these dimensions to the actual defect dimensions.

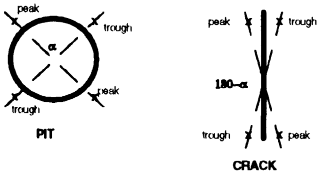


Fig. 3. B*y* peaks and troughs for cracks and pits

3.4 Issues in Real ACFM Sensor Data Interpretation

One of the first problems to arise was that the modelling theory is only fully developed for cracks or notches which are semi-elliptical in shape and hemispherical pits. So for general defects, only approximate classification and characterisation is possible at this stage. This was not perceived to be a major problem. The actual sensor data, however, reproduces theoretically predicted data with varying fidelity. The sensor, however, has a limited resolution depending on the size of the sensor coils, their sensitivity, and other parameters.

The key to the ACFM analysis is the identification of the peaks and troughs of the data. Once these are known, the simplistic rules relating the relative locations of the peaks can be applied for classification. The identification of peaks and troughs is relatively easily done by a human expert; carrying this out automatically requires surface analysis and success depends significantly on the raw data. Noisy data became an obstacle to identifying peaks and troughs and attempts were made to remove noise. The second issue which appears to be a computationally more complex, is the identification of the correct peaks and troughs.

Removal of noise The operation of the electromagnetic sensor is delicate as the sensor is highly sensitive to small deviations in positioning. Hence, real data can be distorted by noise. It was found that for the purposes of the initial system, simple masking was sufficient to remove some of the effects of noise. This allowed underlying peaks and troughs to be more easily identified. For more complex problems of noise, further work may require more complex algorithms from signal processing techniques.

Peak/Trough Location Methods. Three methods were considered for locating the peaks and troughs of the masked B data. The first was a simple search for peaks and troughs: the points with the greatest and least values of B were associated with peaks and troughs. This method was too naive and cannot be relied upon for any but the most regular surfaces.

A more robust method is to locate groups of data points that make up peaks and troughs, or local maxima and minima. The criteria used to find peaks and troughs are: a maximum value is greater than the surface values at all eight adjacent points, and each adjacent point of a maximum has value greater than the values at all adjacent points to it lying further away from the maximum (and likewise for a minimum). This will locate all peaks and troughs, which may still provide inconclusive results in real data if too many are found. Decision rules may be used to select maxima and minima to consider. For example, B*x* and B*z* peaks and troughs should lie along the same line. This search method was chosen for AIRES.

A more sophisticated method which was briefly investigated was that of surface fitting to known expressions, but the difficulty was finding appropriate functions. Further future work could consider advanced methods from signal interpretation.

4. The Blackboard System

The AIRES system was built using TRAM (Tableau noir pour la Robotique Autonome Mobile), which is a multi-expert system development tool originally developed for robotics [5,6]. It has a typical blackboard architecture with *knowledge sources* that are independent modules accessing a common database structure or *blackboard,* with the *controller,* or inference engine, to manage the knowledge sources and the blackboard [7,8]. TRAM was developed in C and thus all KSs and concepts include references to C data structures.

There are several reasons for a blackboard approach to software development [9]. The AIRES project involved several development groups, and the modularity of the blackboard KS structure allows for separate development and testing of code. Moreover, AIRES includes two main separate areas of expertise, electromagnetic and vision, each requiring its own knowledge representation. A blackboard system allows for easy integration of these differing approaches. The only requirement was for the blackboard data structures to be defined early on to be able to communicate across the two subsystems. Other benefits include that they provide dynamic control to allow opportunistic reasoning and an incremental approach to problem solving, and multilevel data to enable reasoning based on data of various levels of granularity.

4.1 Blackboard Concepts

The blackboard can be viewed as an object-oriented database. Automatic interpretation of inspection data requires information on the component being inspected, the sensors employed and their mode of operation, initial sensor data, interpretations of sensor data and data fusion results. Thus the concepts defined for this application have to represent each one of the above. The structure of a concept is shown in Fig 4. It reflects previous findings [10] that three levels of data or information is stored. The lowest level here corresponds to pre-processed sensor data; the mid level stores data on the existence of features, that is peaks and troughs; and finally the top level are characterised defects. The concepts are shown in Fig. 5.

Main structure	a C structure type, accessible only by the KS
inherent attributes	C structure types accessible by both KS and the BBM
attached attributes	C structure types accessible only by the BBM
relation attributes	pointers to other concepts in the blackboard

Program	the name of the KS C function
activation condition	if blackboard database satisfies this condition, the KS is activated
input	the KS C function input parameters
proposition	instructions for the BBM to update the blackboard with the KS results
output	the KS C function output parameters
error test	provides a condition for the error handling to be activated
error action	specifies the particular error action

Fig. 4. TRAM concept and knowledge source cards

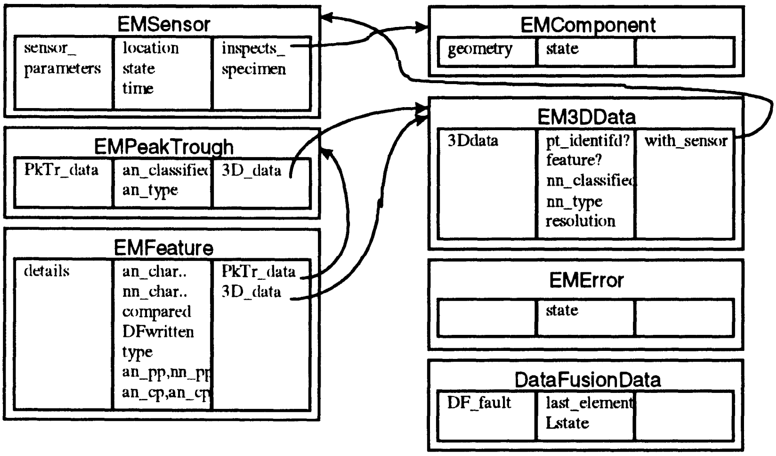


Fig. 5. Electromagnetic sensor KBS concepts

4.2 Knowledge sources

The core of a KS in TRAM is a C function that carries out numerical or other routines. A mechanism is then required to link the C function to the blackboard. The *knowledge source card* (Fig. 4) provides TRAM with a description of when to use the function, required input and the effect that the output from the function will have on the common data storage or blackboard. Fig 6 shows a flowchart of the expected order of activation of the KSs. The KS names indicate they are part of the electromagnetic sensor sub-system ("EM") and if a neural net ("NN") is included.

5. Issues in the Design of Concepts and Knowledge Sources

Many issues and problems related to knowledge representation arose in the first phase of work. Some, such as problem of choosing a scheme for a general inspected object, were easily solved. It was found that a concept representing a flat plate could be easily re-used to model a camshaft component, say, by using polar co-ordinates that map the surface of approximately cylindrical components onto a flat surface. This was sufficient as indeed the sensor probe was designed for inspection of flat or gently curving surfaces and not for sharp corners. Other issues relating to encapsulating the human experts' knowledge were not so easily solved.

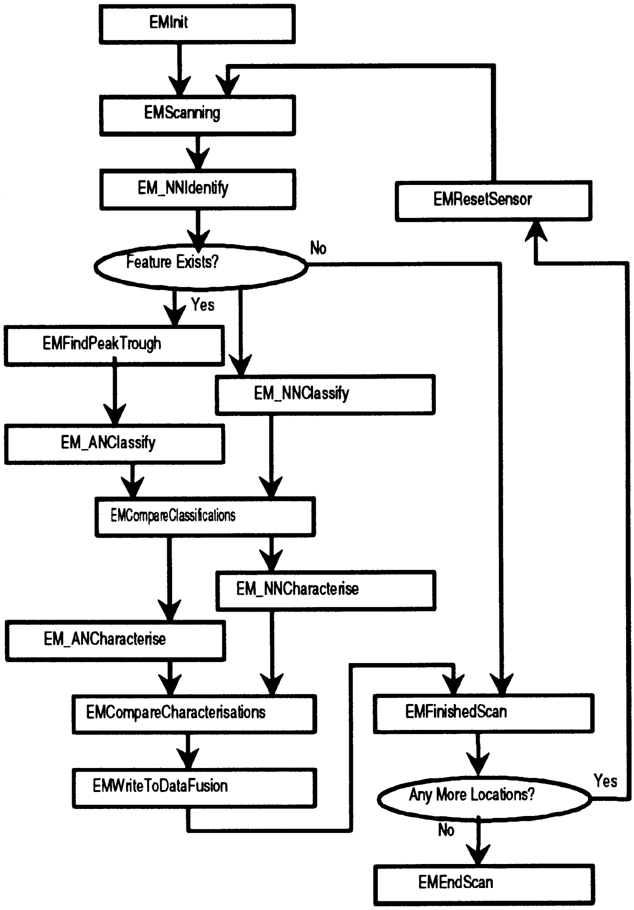


Fig. 6. Flowchart of electromagnetic sensor KS operations

5.1 The Use of Neural Networks

As described earlier, human experts can relatively easily identify flaws from the magnetic data and indeed they believed that it would be relatively simple to encode their reasoning. The KS attempting to encapsulate this expertise was found not to function well. It was found that this step was best performed by a neural network. The function based on human expertise was found to be very much less effective and was removed. This failure may have been due to a lack of understanding of the role of pattern recognition. It was also found that neural nets can also carry out characterisation step in parallel to the KS based on EM theory. Future work would consider pattern matching algorithms and explore further use of neural nets.

5.2 Data Fusion

The complete AIRES system is intended to use the data from the vision and electromagnetic sensors to provide a full defect inspection of a particular component. This involves *data fusion*, whereby data from different sources are somehow combined. In general there are two basic types of data fusion. In early fusion, the data is merged near the sensor, and in late fusion, merging is performed later during computational analysis and processing of the data. Since most of the work concentrated in automating electromagnetic signal interpretation, the main form was late fusion by combining final results of the electromagnetic sensor and the vision KBSs. In addition, some early data fusion takes place between the neural net KSs and the analytical methods KSs in the electromagnetic sensor KBS. The implemented data fusion process is rather simplistic [11].

5.3 Asynchronous Use of Sensors

In the current system, the two sensors do not necessarily scan the component in the same order, and defects may not be detected simultaneously. This is not a problem as long as information on the time of detection and location of the possible defect is stored. If this is the case, then opportunistic inspections can take place. This allows a system that fails to interpret the data for one area to request that the other sensor scans the originating area to provide more data to assist its interpretation. The request can be made even if the vision sensor has no knowledge of another sensor existing. The data fusion becomes an iterative process with the aim being to obtain convergent KBS results. In the event of an intractable inconsistency, the operator may be informed.

5.4 Combined and Irregular Flaws

Opportunistic inspections may also occur in the case of combined or irregular flaws. One example is that of a defect which do not lie at the centre of the scanning area or not aligned to the edge of the matrix. Noisy electromagnetic sensor data may not allow for detection of such a feature. If another sensor detects such a feature, then as part of data fusion, re-scanning may occur to cover the area containing the feature.

There is much difference between inspecting a nearly featureless plate with a single crack and a single pit separated from each other, and heavily corroded surface with numerous flaws. In the former case the magnetic field can be easily analysed; in the latter, isolation of features at the pre-processing stage alone will be a considerable task. Combined features, such as a crack overlaid by a pit, also present a major problem. For a general system, the data fusion design should be extended to allow for re-scanning, split features, and inter-KBS communication at any time.

5.5 Intelligent Assessment

The use of a blackboard architecture may also allow future work to include further expert knowledge on the component being inspected and the likelihood and relative importance of defects at different positions. In addition, a set of common blackboard concepts at all levels allows early overall knowledge to be applied for opportunistic inspections. These two factors would transform the system from being merely analytical to an intelligent KBS.

6. Conclusions

The electromagnetic sensor KBS of the demonstrator AIRES system represents a major step towards understanding how ACFM sensor signal interpretation may be carried out automatically and intelligently.

It was envisaged that knowledge sources based on the electromagnetic theory would be sufficient. It was found, however, that much work was required to smooth the raw data before the algorithms could be applied. Surprisingly, the neural networks created independently of the theory-based KSs were very successful at flaw detection.

Intelligent and early data fusion involving the combining intermediate results from different sensors with knowledge about the inspection techniques and the component being inspected may enhance significantly the total capabilities of a multi-sensor system. Future work in this area, concentrating on the blackboard system approach with multi-level data fusion, may lead to sophisticated automated multi-sensor inspection systems.

Acknowledgements

The work described in this paper was part of the AIRES project which was partly funded by the EC under the BRITE/EURAM programme. The partners were Technical Software Consultants Ltd, University College London, Commissariat a l'Energie Atomique, University of Hanover, Peugeot SA and Deutsche Aerospace SA.

References

1. Engelmore, R, & Morgan, T, *Blackboard Systems,* Addison Wesley (1988)
2. Chin, RT, *Automated Visual Inspection: 1981 to 1987,* CVGIP **41** (1988) 346-381
3. Dover ,WD, Collins, R, *Recent Advances in the Detection and Sizing of Cracks Using Alternating Current Field Measurement,* British Journal of NDT, November (1980)
4. Michael, DH, Lewis, AM, McIver, M, & Collins, R, *Thin-skin Electromagnetic Fields in the Neighbourhood of Surface-breaking Cracks in Metals*, Proc. Royal Society **434** (1991) 587-603
5. Koenig, A, Crochon, E, Antoniakis, M, *TRAM Manual,* LETI, Commissariat a l'Energie Atomique (1990)
6. Koenig, A., Crochon, E., *Temporal Reasoning in TRAM,* Proceedings of the Industrial & Engineering Applications of Artificial Intelligence & Expert Systems IEA/AIE-89 Conference, Tullahoma, US (1989)
7. Nii, HP, *Blackboard Systems: Blackboard Application Systems, Blackboard Systems from a Knowledge Engineering Perspective,* The AI Magazine, August (1986)
8. Nii, HP, *Blackboard Systems: The Blackboard Model of Problem Solving and the Evolution of Blackboard Architectures*, The AI Magazine, Summer (1986)
9. Corkill, D, *Blackboard Systems,* AI Expert, September (1991)
10. Pau, L., *Knowledge Representation Approaches in Sensor Fusion*, Automatica, **25**:2 (1989)
11. Peers, S, Dharmavasan, S, Collins, R, *Advanced Software for Automatic Defect Classification and Characterisation,* in Studies in Applied Electromagnetics and Mechanics 8, IOS Press (1995)

Local Information Processing for Decision Making in Decentralised Sensing Networks

Simukai W. Utete

Robotics Research Group, Department of Engineering Science, University of Oxford,
Oxford OX1 3PJ, U.K.

Abstract. This paper describes consequences of local information processing for decision making in decentralised systems of sensor nodes. In decentralised data fusion systems, nodes take decisions based on information acquired locally. The ability of such nodes to fuse or combine information is linked to network organisation. Earlier work demonstrates a problem of inconsistency which arises given cyclic information flow in decentralised systems where nodes combine global information. This work shows how this inconsistency limits the decision making capabilities of the sensor nodes. Consequences for real-world systems using decentralised processes for decision making in process monitoring, tracking and aviation are discussed.

1 Introduction

This paper addresses two issues:

- how organisation affects the combination of information in decentralised systems of sensor nodes
- how this influences the ability of such sensor nodes to take decisions.

Decentralised information processing and decision making are considered here in the context of sensing. Decision making in decentralised sensing systems has been investigated by earlier authors such as [17,21]. This work considers the implications of a problem of cyclic information flow [30] for decentralised sensor decision makers.

Many sensing systems use multiple sensors, rather than a single sensor, to acquire data about an environment. This approach can promote fault-tolerance, provide a more comprehensive view of an observed environment and allow the combination of different types of data. The main issue in any multi-sensor system is how to combine or fuse diverse sensor data effectively, so that a consistent, coherent and comprehensive view of the observed environment is acquired [11, 20]. Multi-sensor systems typically provide large amounts of data which can be highly redundant and in which conflicts can occur.

In conventional, centralised, data fusion systems (Fig. 1), data from distributed sensors is fused at a central node. The sensors have no ability to process data. Centralisation offers advantages such as simplicity of management. At

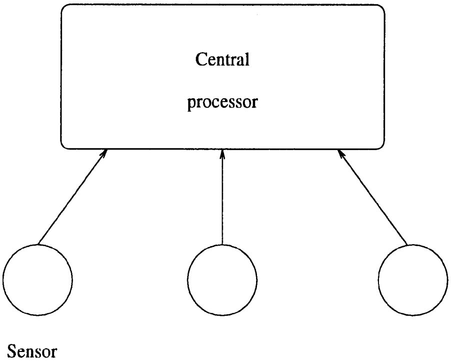


Fig. 1. Centralised sensing system

every stage, the central processor has global information about the state of the system, enabling it to detect redundancies and resolve conflicts in data. Serious drawbacks include the reliance of the system on the correct functioning of a single processor. Centralised processing is also a source of communication and computation bottlenecks. Such drawbacks motivate the study of sensing systems which distribute computation.

1.1 Decentralised Systems

In decentralised data fusion systems [14,25], there is no centralised communication, coordination or control. All computation is distributed to sensor nodes which are capable of processing data, communicating information and taking decisions (Fig. 2). The systems considered in this paper are subject to three constraints on operation [14]:

1. No node is central to network operation.
2. Nodes have only local knowledge of topology.
3. Nodes communicate only locally, with their nearest neighbours.

Maintaining the decentralised constraints leads to many system benefits. The lack of centrality of any node means the system is robust. Nodes are interchangeable, so the system is modular and flexible. As communication is local, bottlenecks are avoided and the system is scaleable. These properties are particularly important in designing autonomous, large-scale and fault-tolerant systems. Decentralised systems have been applied in robotics, surveillance and tracking and process monitoring, for example, in [12,15,17,21,25,26].

The advantages of decentralisation are accrued at some cost: the lack of centralised coordination can lead to inconsistencies between local and global information. The rest of this paper considers the impact of such problems on decentralised decision making. The next section outlines algorithms, based on two types of network topology, which meet the conditions on decentralised information processing [14,25]. Section 3. outlines how the possibility of cycles

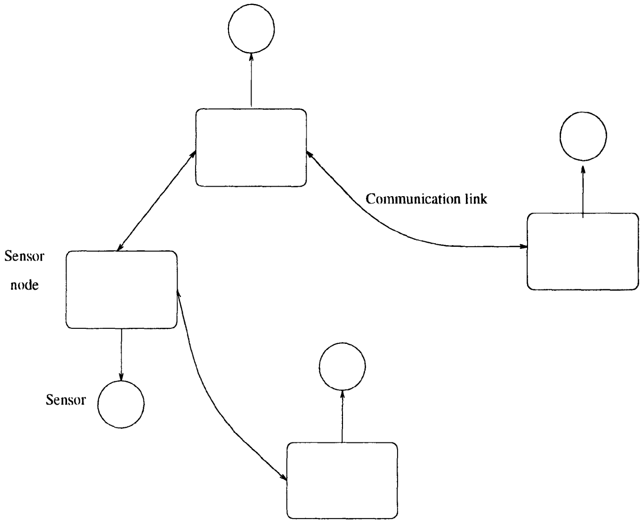


Fig. 2. Decentralised sensing system

in information flow limits the application of decentralised processing to other networks [30]. Sect. 4 demonstrates how this undermines decentralised decision making. Implications for practical systems are considered in Sect. 5.

2 Decentralised Information Processing

Data fusion in the decentralised systems considered in this paper is subject to two conditions.

1. No sensor's information should be incorporated more than once.
2. Each sensor should eventually form a global fused value, based on the information of the entire network.

The two conditions imposed on the network are a safety and a liveness condition whose conjunction must hold [1]. The first relates to the consistency of the fused result. Disproportionate accumulation of data distorts or biases the value a node forms by giving undue weight to a single piece of information. The problem is compounded by communication, which can propagate bias throughout the network, leading to distortion in information fusion at other nodes. Violation of the first condition would lead to an irremediable situation. The second condition indicates that global information must be fused. This is required if the fused result is to be equivalent to that which the corresponding centralised system would produce.

2.1 Common Information

The key challenge in distributed data fusion is identified by Chong et al. [9] as the detection of redundancy. Decentralised sensors may be unable to determine the extent to which their knowledge is shared when there is partial overlap of that information. The complexity of decentralised decision problems is analysed by Tsitsiklis [28]. The problem of partial overlap of communicated information arises in a number of areas, as rumour propagation [10] or double counting [14, 30] in distributed sensing and as correlation in inference networks [22].

In decentralised systems, the problem of common information is exacerbated by the use of local communication. It can be posed in the following "dining table" analogy, due to F. Banda [3]: Conditions at a formal dinner permit guests to communicate with their immediate neighbours, but not to hail distant diners. Therefore, information from remote speakers can be gleaned only through the communication of intermediates. The same piece of information can arrive along both sides of the table. A diner receiving information from two assumed independent sources can arrive at a biased view. The information might be viewed differently if the diner were aware that both communications arose from a single source.

The common information problem is stated here using the following set-theoretic notation derived by Ho [17]. Each node i is associated with a neighbourhood, N_i, the set of nodes to which it is directly connected. Let $[N_i]$ denote the neighbourhood including i or complete neighbourhood, $N_i \cup i$. At each timestep k, node i forms an observation information set $\mathbf{Z}_i{}^k$ based on its local sensor observation, $\mathbf{z}_i(k)$, and on the information communicated by the nodes in its neighbourhood. In other words, the node forms the union of the information sets in the complete neighbourhood $[N_i]$: $\bigcup_{j\in[N_i]} \mathbf{Z}_j{}^k$. For communication between node i and a neighbour j, the required term is

$$\mathbf{Z}_i{}^k \cup \mathbf{Z}_j{}^k = \underbrace{\mathbf{Z}_i{}^k + \mathbf{Z}_j{}^k}_{\text{communicated}} - \mathbf{Z}_{i\cap j}{}^k. \tag{1}$$

This represents the new information that must be assimilated. A fully decentralised solution is possible only where the intersection of communicated information, $\mathbf{Z}_{i\cap j}{}^k$, can be determined from information which is locally available.

2.2 Topology and Common Information

In decentralised systems, the organisation of nodes, or network topology, is central to determining how information is fused. Algorithms satisfying the three constraints and two conditions given have been derived for two types of topology (Fig. 3). Rao [25] develops an algorithm for decentralised data fusion on fully connected networks; In [14], Grime et al. provide an algorithm which can identify common information on tree connected networks. These algorithms are fully described in [25] and [14]. Here it is noted that both can be considered special cases of the common information problem:

1. In the fully connected case, the information common to the communicating nodes is always the same and global. The fully connected condition means that global and local information are in fact the same.
2. In the tree case, there is no overlap in the information communicated between nodes. The algorithm exploits the fact that there is a single communication path between each pair of nodes in a tree.

Both approaches have practical limitations. Fully connected communication poses problems of scale since the number of communication links required increases as the square of the number of nodes. The tree approach is vulnerable since there is no alternate communication path should failures occur.

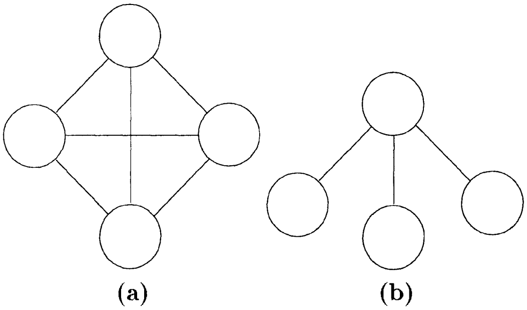


Fig. 3. (a) Fully connected network (b) Tree connected network

3 Inconsistencies in Local Information Processing

The limitations of communicating on fully connected and tree networks motivate the use of other topologies for decentralised information processing. In particular, the ability to operate on a range of networks would allow practical systems to overcome changes such as faults during operation. Local processing of global information on multi-path networks can lead to the decision problem represented by the following analogy.

3.1 The Successor Problem

The king of B– faces a problem of succession. The order of succession has traditionally been from the king to his eldest son. Successive kings have been killed by impatient offspring. Finally, one king resolves to change the succession. He declares that succession will henceforth be decided by a council of elders, sons and other interested parties. Each member of the council will be able to confer with two others, but the full composition of the council will be known only to the monarch. All communication will be anonymous. The decision process must not terminate before everyone's view is considered and it must not count anyone's views more than once. [1] The king tells his sons that the first one able to determine the decision of the council will be his successor. The process of negotiation begins. At each stage, the sons communicate with their neighbours, the neighbours communicate with their neighbours and so on around the circle. At no point does a son have sufficient information to end negotiation by declaring himself king.

3.2 The Cycle Problem

The problem of the kings of B– arises given decentralised information processing on cycles. This problem is analysed in earlier work [30,31]. Results from that work are re-stated here and shown in the next section to undermine the decision process.

In [30,31], a result from automata theory [27] is shown in to have implications for distributed data fusion. In [27], Rosenstiehl et al. prove a number of graph properties undecidable for automata using local or myopic algorithms in symmetric networks. Of particular relevance, they show that an automaton using a deterministic local algorithm cannot determine network connectivity in a symmetric graph. The crux of the problem is that the automaton cannot decide whether a link connects its subgraph to another, which is identical, or to a single automaton.

In common with networks of automata, decentralised networks use a local communication strategy and some of these problems follow. However, important distinctions arise in the decentralised case. The derivation of a data fusion algorithm for a decentralised network does not require local determination of the network's connectivity and network symmetry does not necessarily prove an exhaustive consideration. This is demonstrated by the algorithm of Grime [13] which makes the assumption that the network is tree connected, but operates on any tree, symmetric or otherwise. However, the circumstances which lead to undecidable connectivity cause problems of network construction. The result of [27] has the following consequence for decentralised data fusion systems:

Lemma 1. *Any cycle network G can be used to create a larger cycle G' containing m copies of G, where $m \geq 2$, and in which the same local conditions hold.*

The problem is that local nodes can fail to distinguish networks which are locally isomorphic in structure yet have global differences, for example in size. This uniformity of local conditions arises even with non-deterministic algorithms. The cycle problem has serious consequences for distributed data fusion. It is shown [30] that:

Theorem 1. *For each integer $n \geq 4$, there exists a graph G on nodes $1, ..., n$ and a sequence of inputs such that there is no algorithm which, on the basis*

[1] This problem of succession resolved by council has arisen (BBC radio documentary, 1997), the resolution in that case being a non-cyclic, non-decentralised one.

of local node conditions alone, can satisfy simultaneously the properties that all information is fused in finite time and none is fused more than once.

It is not possible to generalise A to all connected graphs intermediate to tree and fully connected graphs. In structures which are cyclic, there is no locally available information which would enable a sensor node to combine information globally and without redundancy within the decentralised constraints.

4 Consequences for Decentralised Decision Making

Decisions derived by decentralised processes can be undermined by the inconsistencies seen with local information processing.

4.1 The Cycle Problem Revisited

The organisation of the decentralised network determines how information is presented to the sensor nodes. Problems posed on cycles force the consideration of information using a pairwise strategy or binary decision processes. In the successor problem analogy, by insisting on the council being constituted as a cycle in which participants have only local knowledge, the king ensures that binary decisions are used by the local participants in the decision making process. Only an external (centralised) observer or more knowledgeable participant (one with pertinent global information) can solve the successor problem, even though all participants eventually gain access to the requisite information. Binary decision making processes have been investigated extensively in the context of voting [2] and decisions derived from them can be susceptible to other inconsistencies such as order dependency. Such problems relate to how the information is grouped prior to taking a decision and can preclude centralised equivalence of a decentralised decision. This suggests methods of solution based on controlling information flow.

Suboptimal strategies have been suggested by a number of authors as a means of dealing with redundant information in distributed data fusion systems. These methods relax the decentralised constraints. For global fusion systems, strategies include: distinguishing a node or nodes [27], widening node knowledge of topology [14], communicating more information, for example, by tagging data [14], organising information using clustering strategies from Artificial Intelligence [22] as in [30]. The concern here is with problems of organisation of information flow. Organisation breaks with the decentralised constraints by permitting groups of nodes greater importance in the network. This gives an order of importance to nodes in the network, effectively breaking cycles. A method suggested by Grime [14] for decentralised operation in multi-path structures is to break links, generating spanning trees for communication of information. An implementation of link breaking which uses dynamic programming to generate trees is described in [30,31] and outlined below.

5 Applications

This section considers three applications where cycles of information flow can arise.

5.1 Decentralised Management of a Process Plant

An experimental process monitoring system, of industrial scale and complexity, is described in [12]. The plant comprises over 200 distributed sensors and over 20 decentralised sensor nodes. Its standard configuration is a tree network. In order to achieve greater robustness, a multi-path network is required. This necessitates compromise of the decentralised constraints.

Link breaking, as suggested in [14], is applied. The implementation of [30] compromises constraints in only one respect. It uses an asynchronous implementation of the Bellman-Ford dynamic programming algorithm [6] derived by Bertsekas [7, 8] to generate spanning trees. The dynamic programming algorithm requires the distinction of a node as the global destination of the tree. Applying this algorithm reduces the cost of suboptimal operation from that of organising the entire network to the distinction of a single node as the global destination of the spanning tree.

5.2 Implications for Other Applications

The problem of decentralised information processing on cycles suggests implications for decision fusion in other practical applications: In data association, the objective is to relate tracks to the targets which generate them. In multi-target problems, two or more tracks may compete for the same point if their observation information sets overlap [4, 26]. Drawing from the work presented here, if decentralised resolution of conflict between overlapping information sets generates cycles this can lead to the problem of undecidability. At the same time, decision making which does not take global information into account can result in inconsistency.

A recent proposal of the US Federal Aviation Authority is the "free flight" programme for air traffic management [23]. This will effectively decentralise air traffic control by allowing individual aircraft greater autonomy in route planning and decision making. (The author's attention was drawn to this problem by J. Uhlmann [29].) The cycle problem would have implications for free flight if decision makers truly were fully decentralised as defined above. The problem of partially overlapping information generating inconsistencies can be avoided by using global information in aircraft information processing and decision making. If fully decentralised processing were applied, on-line changes to network configuration would be possible and cycles in information flow could arise.

6 Conclusions

This paper considers decision problems arising from decentralised information processing. Decentralised systems are highly suited to real-world applications where scale is important or fault tolerance a priority. Understanding the impact of decentralised information processing on decision making is necessary in designing practical systems, particularly safety-critical ones.

Consistency problems arise in many domains where local decisions are combined to generate global outcomes. In distributed data fusion problems, a major challenge is redundancy and associated bias. Earlier work demonstrates that failure to detect redundancy leads to undecidability when fusion occurs on cyclic structures. This undermines decentralised decision making: even when the information required is available, a decentralised process can fail to reach a decision to combine it consistently. This suggests that, in decentralised decision making, the organisation of information flow can be as important as the information itself. Implications of decision problems presented in the paper are suggested for some real-world applications.

Acknowledgements

Work on decentralised information processing was done in the context of a doctoral degree supervised by Dr. H. F. Durrant-Whyte, who provided valuable comments. In formulating the local communication problem, the author benefited from discussions with Dr. A. Stevens and Professor D. Welsh.

The dining table analogy was suggested by Dr. F. Banda. J. Uhlmann drew the author's attention to the free flight application.

The author acknowledges with thanks the support of the Lady Wolfson Junior Research Fellowship in Engineering Science at St. Hugh's College, Oxford, in carrying out this work.

References

1. Alpern, B., Schneider,F.: Defining Liveness. Information Processing Letters, Vol. 21. No. 4. (1985) 181-185
2. K. J. Arrow.: Social Choice and Individual Values. Wiley, New York (1951)
3. Banda, F.: Personal communication (1995)
4. Bar-Shalom,Y., Fortmann, T.E.: Tracking and Data Association. Academic Press (1988)
5. Bell D.E., H. Raiffa, H., A. Tversky, A.: Decision Making; Descriptive, Normative and Prescriptive Interactions. Cambridge University Press (1988)
6. Bellman, R.: Dynamic Programming. Princeton University Press (1957)
7. Bertsekas, D.P.: Distributed Dynamic Programming. IEEE Transactions on Automatic Control. Vol. 27 No. 3 (1982) 610-616
8. Bertsekas, D.P.: Distributed Asynchronous Computation of Fixed Points. Mathematical Programming Vol. 27 (1983) 107-120
9. Chong,C., Mori, S., Chang, K.: Distributed Multitarget Multisensor Tracking. In: Bar-Shalom, Y., (ed.): Multitarget-Multisensor Tracking: Advanced Applications. Artech House, Norwood, MA, (1990) 247-295
10. Chong, C-Y., Liggins II, M.: Fusion Technologies for Drug Interdiction. Proceedings of the IEEE Conference on Multisensor Fusion and Integration for Intelligent Systems. (1994) 435-441

11. Durrant-Whyte, H.F.: Consistent integration and propagation of disparate sensor observations. The International Journal of Robotics Research (1987) 6(3) 3-24
12. Gao, Y., Durrant-Whyte, H.F.: A Transputer Based Sensing Network for Process Plant Monitoring. Proceedings of the Third International Conference on Transputers and Applications. IOS Press. 180-185, (1991)
13. Grime, S.: Communication in Decentralised Sensing Architectures, Ph.D. Thesis, Robotics Research Group, Department of Engineering Science, University of Oxford (1992)
14. Grime, S., Durrant-Whyte, H.F., Ho,P.: Communication in Decentralised Data Fusion Systems, Oxford University Engineering Lab, Technical Report No. 1900/91, Oxford University Department of Engineering Science (1991)
15. Grime, S., Durrant-Whyte, H.F.: Data Fusion in Decentralized Sensor Networks. Control Engineering Practice. Vol. 2. No. 4, Pergamon Press 849-863 (1994)
16. Harary, F., Norman, R., Cartwright, G.: Structural Models: An Introduction to the Theory of Directed Graphs. Wiley (1965)
17. Ho, P.: Organisation in Distributed Sensing. Ph. D. Thesis. Robotics Research Group, Department of Engineering Science. University of Oxford (1995)
18. Ho, Y.C.: Team Decision Theory and Information Structures Proceedings of the IEEE. Vol. 68 (1980)
19. Kim, K.H., Roush, F.W.: Introduction to Mathematical Consensus Theory. Marcel Dekker, Inc., Lecture Notes in Pure and Applied Mathematics, Vol. 59. (1980)
20. Llinas, J., Waltz, E.L. (eds.): Multisensor Data Fusion. Artech House, Norwood, MA (1990).
21. Manyika, J., Durrant-Whyte, H.F.: Data Fusion and Sensor Management: a Decentralized Information-Theoretic Approach. Ellis Horwood, UK (1994)
22. Pearl, J.: Probabilistic Reasoning in Artificial Intelligence. Morgan Kaufmann (1988)
23. Perry, T.S.: In Search of the Future of Air Traffic Control. IEEE Spectrum. (August 1997)
24. Radner, R.: Hierarchy: The Economics of Managing. Journal of Economic Literature. Vol. XXX 1382-1415 (September 1992)
25. Rao, B.S.Y., Durrant-Whyte, H.F.: Fully Decentralised Algorithm for Multisensor Kalman Filtering. IEE Proceedings-D. Vol. 138. No. 5. (1991) 413-420
26. Rao, B.S.Y., Durrant-Whyte, H.F.: A Decentralized Bayesian Algorithm for Identification of Tracked Targets, IEEE Transactions on Systems, Man and Cybernetics. Vol. 23. No. 6. (1993) 1683-1697
27. Rosenstiehl, P., Fiskel, J.R., Holliger, A.: Intelligent Graphs: Networks of Finite Automata Capable of Solving Graph Problems. In: Read, R.C. (ed.) Graph Theory and Computing. Academic Press (1972) 219-265
28. Tsitsiklis, J.N., Athans, M.: On the Complexity of Decentralised Decision Problems. IEEE Trans. on Automatic Control Vol. 30, No. 5 (1985)
29. Uhlmann, J. Personal communication. (1997)
30. Utete, S.W.: Network Management in Decentralised Sensing Systems. Ph. D. Thesis, Robotics Research Group, Dept. of Engineering Science, University of Oxford, Oxford, U.K. (1995)
31. Utete, S.W., Durrant-Whyte, H.F.: Local Communication Constraints in Decentralised Sensing Networks. IEEE Trans. on Systems, Man and Cybernetics. (Submitted for review, May 1997)

A Fuzzy Linear Approach to Recognising Signal Profiles *

P. Félix[1], S. Fraga[1], R. Marín[2], and S. Barro[1]

[1] Departamento de Electrónica e Computación
Facultade de Física. Universidade de Santiago de Compostela
{paulo,santi,senen}@dec.usc.es
[2] Facultad de Informática. Universidad de Murcia
roque@dif.um.es

Abstract. In other papers we have introduced the model of Fuzzy Temporal Profile (FTP) [4] through which we describe the evolution of a particular physical parameter V over time. Following this model an approximation to the evolution curve by means of sections between a set of significant points $(X_0, X_1, ..., X_n)$ is defined. Each section is defined by way of an imprecise constraint on duration, on increase in value and on slope between the points connected by the section.
In a previous paper [5] we presented a method of matching which analyses the occurrence of certain signal events associated to a particular temporal instant, given by a fuzzy predicate, and a certain condition of value, given, in the same manner, by a fuzzy predicate. In this paper we go beyond this, tackling the following problem: given an FTP and a signal representing the evolution of a certain system, to find the degree of compatibility between the profile and a piece of this signal, and the most possible locations in the global evolution.

1 Introduction

Any approach which studies the behaviour of a system does so by way of certain characteristic features in the evolution of some of its representative variables. When we do not have an exact model which identifies its behaviour, a process of interpretation is followed which aims to relate this behaviour with some of its signs, which are often identified with profiles which an expert forms mentally on its evolution, and which he uses to compare with real data. We will try to model these profiles by means of a fuzzy model which we call Fuzzy Temporal Profile (FTP).

The FTP model is based on the use of the formalism of constraint networks (CN). On one hand, CN's enable an immediate projection of the linguistic description of a profile that is of interest. Besides, they constitute an adequate support for the analysis of the information which it is necessary to present, be

* This work has been funded by the Comisión Interministerial para la Ciencia y la Tecnología (CICYT), through the research project TIC95-0604.

this redundant or contradictory. On the other hand, we model the necessary flexibility in the associated constraints on a profile by means of the fuzzy set theory [3]. In this way the model affords us a way of analysing the problem of the representation of tendencies in imprecise terms, closer to those which form part of the criteria of human experts.

We have organised the article in the following manner: firstly we present the fundamental concepts of the constraint network model which we call FTP. We then go on to define the semantics of the concept of section in which the method of matching we propose to use is based, together with the matching procedure itself. Lastly we analyse the problem of the location of a profile on the global signal.

2 Definition of the model

2.1 Time

We are going to consider time as projected on a unidimensional discrete axis $\tau = \{t_0, t_1, ..., t_i, ...\}$. Thus given an i belonging to the set of natural numbers $\mathbb{N}$, t_i represents a *precise instant.* We consider a total order relation between the precise instants $(t_0 < t_1 < ... < t_i < ...)$, and a uniform distance between them, in such a way that for every $i \in \mathbb{N}$, $t_{i+1} - t_i = \Delta t$, Δt being a constant. This constraint corresponds to that of periodic signal sampling, which is fairly frequent in real applications.

2.2 Initial definitions

Taking Zadeh's extension principle as a starting point [8] below we will introduce the concepts of *fuzzy value* and *fuzzy increase*, on which the bulk of the model's concepts are based.

Definition 1. *Given as universe of discourse $\mathbb{R}$ or $[m,n] \subset \mathbb{R}$ we extend the concept of value to what we will call fuzzy value c, represented by a possibility distribution π_c over $\mathbb{R}$. This way, given a precise value $v \in \mathbb{R}$, $\pi_c(v) \in [0,1]$ represents the possibility of c being precisely v. We will always assume that π_c is normalised, that is, $\exists v \in \mathbb{R}$, $\pi_c(v) = 1$; and unimodal, that is: $\forall v, v', v'' \in \mathbb{R}$, $v < v' < v''$, $\pi_c(v') \geq min\{\pi_c(v), \pi_c(v'')\}$.*

Definition 2. *We introduce the concept of fuzzy increment in order to represent amounts, such as, for instance, the difference between two values. A fuzzy increment d is represented by means of a normalised and unimodal possibility distribution π_d, which is also defined over $\mathbb{R}$. This way, $\forall \delta \in \mathbb{R}$, $\pi_{D(a,e)}(\delta) = \sup_{\delta=t-s} min\{\pi_a(s), \pi_e(t)\}$*

2.3 Fuzzy Temporal Profiles

In other paper [4] we defined a model, which we named *Fuzzy Temporal Profile* (FTP), which carries out a fuzzy linear description of the evolution of a physical parameter. An FTP is a network of fuzzy constraints between significant points, which perform the role of network nodes. Each significant point is defined as a pair of variables: one corresponding to the value of the physical parameter $v(t)$ and the other to time. The fuzzy profile constraints limit the fuzzy duration, the fuzzy increase and the fuzzy slope between each of the significant points.

A fuzzy representation of the constraints enables the expression of flexible evolution profiles, besides formalising in a direct and simple way concepts such as *preference* between solutions, that is to say that certain solutions are considered as being more acceptable than others; or the concept of *priority* between constraints [2], in such a way that these relevant constraints have priority over those which are considered to be of little importance.

We now go on to define the fundamental concepts of the model.

Definition 3. *We define significant point associated with a variable $v(t)$, and this is denominated as X_i, as the pair formed by a variable of the domain V_i, and a temporal variable T_i.*

$$X_i =< V_i, T_i > \tag{1}$$

where V_i represents an unknown value of the physical parameter, and T_i represents an unknown time instant. In the absence of any constraints, the variables V_i and T_i may take any value, be it of a crisp or fuzzy nature.

Definition 4. *A unitary constraint L_i on a temporal variable T_i is defined by means of a normalised and unimodal possibility distribution $\pi_i^L(t)$, whose discourse universe is the time axis τ. $\forall\ t \in\ \tau :\ \pi_i^L(t)\ \in\ [0,1]$, so that given a precise time instant t_i, $\pi_i^L(t_i)$ represents the possibility that T_i takes precisely the value t_i.*

The unitary constraint L_i restricts the domain of values which may be assigned to T_i to those time instants t_i which satisfy $\pi_i^L(t_i) > 0$. Formally, the distribution $\pi_i^L(t)$ corresponds to the possibility distribution of a fuzzy value. Thus we can interpret a unitary constraint L_i, as the assignment of a fuzzy value, which we call *fuzzy instant*, to the variable T_i.

Definition 5. *A unitary constraint D_i on a variable of the domain V_i is defined, in a similar way to L_i, by means of a normalised and unimodal possibility distribution $\pi_i^D(v)$, whose universe is $\mathbb{R}$. $\forall\ v \in\ \mathbb{R} :\ \pi_i^D(v)\ \in\ [0,1]$*

We may interpret a unitary constraint D_i as the assignment of a *fuzzy value* to the variable V_i.

Definition 6. *A binary constraint L_{ij} on two temporal variables T_i and T_j is defined by means of a normalised and unimodal possibility distribution π_{ij}^L, whose discourse universe is $\mathbb{Z}$.$\forall\ d \in\ \mathbb{Z} :\ \pi_{ij}^L(d)\ \in\ [0,1]$ Given a precise value d_{ij}, $\pi_{ij}^L(d_{ij})$ represents the possibility that the temporal distance between T_i and T_j take precisely the value d_{ij}.*

The constraint L_{ij} jointly restricts the possible value domains of the variables T_i and T_j. In the absence of other constraints, the assignments $T_i = t_i$ and $T_j = t_j$ are possible if $\pi^L_{ij}(t_j - t_i) > 0$ is satisfied. Formally, the distribution π^L_{ij} corresponds to the possibility distribution of a fuzzy increase. Thus, we may interpret a binary constraint L_{ij} as the assignment of a fuzzy increase, which we call *fuzzy duration*, to the distance between the variables T_i and T_j.

Definition 7. *A binary constraint D_{ij} on two variables from the domain V_i and V_j is defined, in a similar way to the constraint L_{ij}, by means of a normalised and unimodal possibility distribution π^D_{ij}, whose discourse universe is $\mathbb{R}$.* $\forall\, \delta \in \mathbb{R}: \ \pi^L_{ij}(\delta) \ \in \ [0,1]$

We may interpret a binary constraint D_{ij} as the assignment of a *fuzzy increase* to the distance between V_i and V_j.

Definition 8. *A quaternary constraint M_{ij} on two significant points X_i and X_j, is defined by means of a normalised and unimodal possibility distribution π^M_{ij}, whose discourse universe is $\mathbb{R}. \forall\, p \in \mathbb{R}: \ \pi^M_{ij}(p) \ \in \ [0,1]$*

Given a precise value p_{ij}, $\pi^M_{ij}(p_{ij})$ represents the possibility that the slope of the line that joins X_i and X_j be precisely p_{ij}. The constraint M_{ij} jointly restricts the domains of V_i, V_j, T_i and T_j. In the absence of other constraints, the assignments $V_i = v_i$, $V_j = v_j$, $T_i = t_i$ and $T_j = t_j$ are possible if $\pi^M_{ij}((v_j - v_i)/(t_j - t_i)) > 0$ is satisfied. We can interpret a constraint M_{ij} as the assignment of a fuzzy value, which we denominate *fuzzy slope*, to the line which joins X_i and X_j.

Definition 9. *A fuzzy constraint R_{ij} on two significant points X_i and X_j is a trio formed by a fuzzy duration L_{ij}, a fuzzy increase D_{ij}, and a fuzzy slope M_{ij}.*

$$R_{ij} =< L_{ij}, D_{ij}, M_{ij} >$$

Definition 10. *We define a* **Fuzzy Temporal Profile** *(FTP) as a finite set of significant points $\{X_1, X_2, ..., X_n\}$, and a finite set of unitary $\{L_i, D_i\}_{i=1,...,n}$, binary $\{L_{ij}, D_{ij}\}_{i,j=1,...,n}$ and quaternary $\{M_{ij}\}_{i,j=1,...,n}$ constraints defined on the variables which constitute these points. An FTP may be represented by way of a directed graph (figure 1), in which the nodes correspond to significant points, and the arcs correspond to the constraints on the variables of the nodes which they join.*

We will add an additional significant point $X_0 =< L_0, D_0 >$, which represents a precise origin for the time and value axes. We can assign an arbitrary value to this point, but for the sake of simplicity, we assign it the value $L_0 = 0, D_0 = 0$. In this way, $L_{0i} = L_i, D_{0i} = D_i$.

The definition of an FTP may arise from the projection, in a set of constraints between significant points, of a linguistic description on a determined profile of interest made by an expert. This description may contain redundant or inconsistent information, and its analysis is partially dealt with in other works [4].

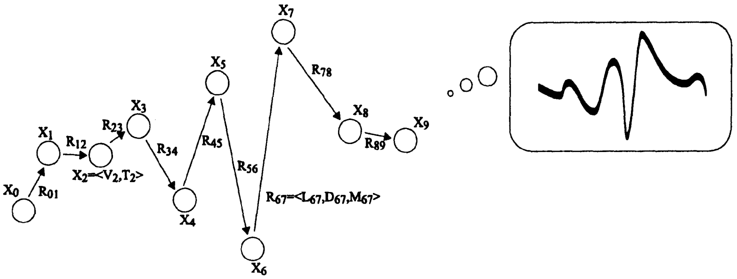


Fig. 1. Example of Fuzzy Temporal Profile. In order to make the figure intuitive we have depicted the constraint network with a sketch of the profile morphology.

3 FTP-signal matching through fuzzy linear sections

The recognition of signal profiles is carried out on the evolution of a particular especially significant physical parameter of the system under study. We assume that this signal is obtained by way of a sampling process which is represented as $P = \{(v_m, t_m),\ m = 1, ..., M\}$.

We aim to find a membership function $\mu_{ij}^{R}(v,t)$ for each one of the segments between consecutive significant points which make up the profile. In this way the FTP becomes a fuzzy linear approximation to the dynamics of the system. In order to obtain the membership function we use the relative information associated to each section -R_{ij}- and the absolute information associated to the starting and finishing points of the section -R_{0i} y R_{0j}-. Thus, by means of a fuzzy extension on the expression of the straight line $y - y_1 = m \times (x - x_1)$, we obtain a fuzzy constraint on the points of the plane (v,t) belonging to the temporal interval between the two significant points.

The extension of the equality is realised by means of Zadeh's extension principle. Thus the truth value of the proposition $A = B$ is:

$$v(A = B) = \max_{s=t} \min\{\mu_A(s), \mu_B(t)\} = \max_{u}\{\mu_{A\cap B}(u)\}$$

This value is the complementary one of the one defined by Zadeh for the separation of two fuzzy sets $D = 1 - M_{A\cap B}$; $M_{A\cap B}$ being the maximum degree of membership of the set intersection between A and B.

The fuzzy constraint given by a linear equation, in the sense given by Bellman and Zadeh[1], assigns a fuzzy set to a fuzzy equality, whose membership function is :

$$\mu_{ij}^{S}(v,t) = \max_{u}\{\mu_{v\ominus D_{0i}\cap M_{ij}\otimes(t\ominus L_{0i})}(u)\} \tag{2}$$

that is to say, we can understand this linear equation as a conjunction of two given constraints, one on the spatial part and the other on the temporal part of the same entity be: one point (v,t) of the space $\mathbb{R} \times \tau$. The previous equation

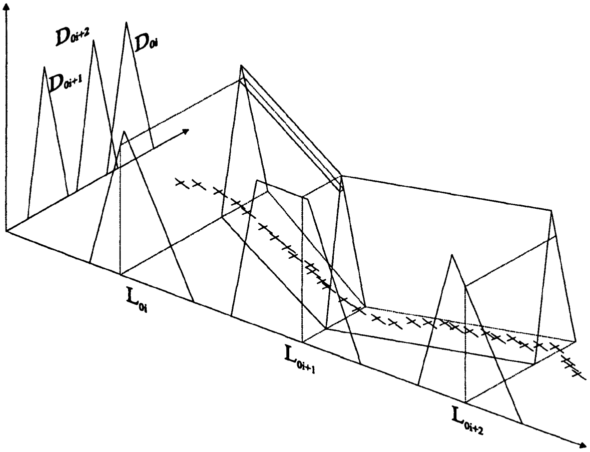


Fig. 2. Example of profile matching by means of a fuzzy linear approximation of each one of its sections. This matching aims for the maximum compatibility between the signal values and the fuzzy sections.

evaluates the degree of membership of a point (v, t) to the fuzzy straight line given by the constraints of the section R_{ij}.

The matching between the profile and a real evolution is realised by means of the assignment to the significant points, in such a way we extract information of the real evolution which we then contrast with the constraints of the network.

Definition 11. *We give the name $\tilde{A}_i$ to the assignment of values for the variables of the significant point $X_i =< \tilde{V}_i, T_i >$, in such a way that a fuzzy value is assigned to the variable V_i, and a precise value to the variable T_i. $\tilde{A}_i =< \tilde{v}_i, t_i >$, means that : $V_i = \tilde{v}_i \leftarrow \pi_i^V(v)$ $\forall v \in \mathbb{R}$ and $T_i = t_i$.*

Definition 12. *We say that a pair of assignments $(\tilde{A}_i, \tilde{A}_j)$ is valid under the constraint R_{ij} or R_{ij}-***valid** *if, and only if, it satisfies the constraint R_{ij} existing between the significant points X_i y X_j.*

$$\pi_{ij}^L(t_j - t_i) > 0$$

$$\pi_{ij}^D(\tilde{v}_j \ominus \tilde{v}_i) \triangleq \Pi_{(\tilde{v}_j \ominus \tilde{v}_i) \circ D_{ij}} = \max_{\delta} \max_{\delta = v_j - v_i} \min\{\pi_i^V(v_i), \pi_j^V(v_j), \pi_{ij}^D(\delta)\} > 0$$

$$\pi_{ij}^M((\tilde{v}_j \ominus \tilde{v}_i)/(t_j - t_i)) \triangleq \max_{\delta} \max_{\delta = v_j - v_i} \min\{\pi_i^V(v_i), \pi_j^V(v_j), \pi_{ij}^M(\delta/(t_j - t_i))\} > 0$$

Definition 13. *We give the name Q_{ij} to the fuzzy set of R_{ij}-valid assignments $(\tilde{A}_i, \tilde{A}_j)$ defined using the possibility distribution:*

$$\pi_{ij}^Q(\tilde{A}_i, \tilde{A}_j) = \min\{\pi_{ij}^L(t_j - t_i), \pi_{ij}^D(\tilde{v}_j \ominus \tilde{v}_i), \pi_{ij}^M((\tilde{v}_j \ominus \tilde{v}_i)/(t_j - t_i))\}$$

The degree of possibility that a pair of assignments $(\tilde{A}_i, \tilde{A}_j)$ are R_{ij}-valid is given by the minimum of the degrees with which it satisfies the time, value and slope constraints.

In the specific problem which we are dealing with, we take as possible assignments to the significant points X_i and X_j, different pairs $\tilde{A}_i = (D_{0i}, t^i) = (D_{0i}, t_m)$, $(v_m, t_m) \in P$, and $\tilde{A}_j = (D_{0j}, t^j) = (D_{0j}, t_n)$, $(v_n, t_n) \in P$ compatible with the constraints of the section. In this case the contents of the following lemma are demonstrated:

Lemma 1. *Given the assignments $\tilde{A}_i = (D_{0i}, t^i)$ and $\tilde{A}_j = (D_{0j}, t^j)$, the degree of possibility that they be R_{ij}-valid assignments is given by the following expression:*

$$\pi^Q_{ij}(\tilde{A}_i = (D_{0i}, t^i), \tilde{A}_j = (D_{0j}, t^j)) = \pi^L_{ij}(t^j - t^i)$$

What is the significance of an assignment of this type? Defining a membership function for each section as a straight line in which the fuzzy value of the beginning of the line (D_{0i}) is projected in each time instant in a direction given by the fuzzy direction M_{ij}. The temporal constraints of the profile serve to delimit the duration of the sections.

In order to calculate the degree of matching between a profile and a fragment of signal we obtain in an incremental way a solution for the FTP, partially calculating its degree of compatibility. In order to do this we realise an assignment to each one of the significant points in the temporal order in which they are defined, and for each two consecutive points the signal samples between these assignments are matched with the corresponding section, according to the following expression:

$$\mu^S_{ij}(\tilde{A}_i, \tilde{A}_j) = \min_{t^i \leq t_m \leq t^j} \max_{u} \{\mu_{(v_m \ominus D_{0i}) \cap M_{ij} \otimes (t_m \ominus t^i)}(u)\} \tag{3}$$

Thus, we say that an ordered k-tuple of assignments $(A_0, \tilde{A}_1, ..., \tilde{A}_k)$ is **locally valid** if, and only if, it satisfies:

$$\mu^{FTP}_{0,...,k}(A_0, \tilde{A}_1, ..., \tilde{A}_k) = \mathop{min}_{\substack{\tilde{A}_i, \tilde{A}_j \in (A_0, \tilde{A}_1, ..., \tilde{A}_k) \\ f=e+1,\ \tilde{A}_e, \tilde{A}_f \in (\tilde{A}_0, \tilde{A}_1, ..., \tilde{A}_k)}} \{\pi^L_{ij}(t^j - t^i), \mu^S_{ef}(\tilde{A}_e, \tilde{A}_f)\} > 0$$

$\mu^{FTP}_{0,...k}(A_0, \tilde{A}_1, ..., \tilde{A}_k)$ being the degree of consistency of this assignment. We give the name $FTP_{0,...,k}$ to the fuzzy set which locally describes the profile in the temporal interval defined by L_{1k}.

Thus the matching function consists of a set of fuzzy straight line sections the duration of which is given by the constraint L_{ij}. Given that each constraint poses a disjunction of possibilities, matching consists of looking for the set of section durations which enables a better adjustment of the set of signal samples to the FTP.

We put forward a method for the search of a global solution based on an assignment tree, initially gone through in depth [3]. At each step for the obtention of global compatibility between the signal and the FTP we extend a k-tuple $(A_0, \tilde{A}_1, ..., \tilde{A}_k)$ of locally valid assignments to the following significant point X_{k+1}. If a pair (D_{0k+1}, t_m) exist, so that the local consistency $\mu^{FTP}_{0,1,...,k+1}(A_0, \tilde{A}_1 ..., \tilde{A}_{k+1}) > c_{\text{inf}}$, being c_{inf} a prefixed lower bound, then we assign (D_{0k+1}, t_m) to the significant point X_{k+1}. If no pair is found that satisfies the previous condition, we backtrack to the assignment of the previous significant point X_k. When we find a global assignment $(A_0, \tilde{A}_1, ..., \tilde{A}_N)$ whose consistency is greater than c_{inf},this will be the current best solution. Then we update $c_{\text{inf}} = \mu^{FTP}_{0,1,...,N}(A_0, ..., \tilde{A}_N)$ and we go back in search of a better solution. All the solutions whose consistency is greater than a given c_{sup} will be considered as good solutions.

Some improvements in the efficiency of the assignment process can be incorporated. Thus we realise the assignments to each significant point by the order of the degree of compatibility. Furthermore, we should bear in mind that if a partial assignment $(\tilde{A}_0, ..., \tilde{A}_k)$ $k < N$ maintains the same degree of compatibility when we extend it to a global assignment, all the possible extensions starting from the significant point X_k can be obviated.

3.1 Location of the profile

The procedure which we have just described for the matching between the signal and the profile aims to maximise the degree of global compatibility, while it does not have compensatory mechanisms: a degree of compatibility which is locally poor is not compensated with another degree of compatibility which is locally good. This fact leads to the theoretical need on cover the whole signal in the matching process. In practice it only leads to a computationally excessive problem.

In numerous problems of the detection of profiles we find that some of the characteristics of the profile are especially significant in the detection. Not only are the more relevant than the rest -a characteristic which can be controlled by way of the concept of priority, previously discussed-, but they also enable a quicker location of the profile on the signal set. One way of benefiting from this characteristic is maximising the degree of compatibility of a section or of a set of sections with the signal. In order to do this we carry out the aforementioned procedure for the sub-profile which defines the feature of interest, completing the location of the entire profile by means of the propagation of constraints starting from the set of assignments with a greater degree of compatibility.

Locating an especially significant section involves searching the signal for a set of samples which verify the conditions of duration, increase in value and slope specified in the evolution given by $\mu^S_{ij}(\tilde{A}_i, \tilde{A}_j)$.

The following expression realises the calculation of the compatibility between the profile and the signal, assigning to each sample $p_m = (v_m, t_m)$ a degree of membership to a section of slope given by M_{ij}, of a duration given by L_{ij}, and a started value given by D_{0i} (see figure 3):

$$\mu_{ij}^{FTP}(p_m) = \max_{d} \min\{\pi_{ij}^{L}(\delta), \max_{d=t^j-t^i} \min_{t^i \leq t_m \leq t^j} \max_{u}\{\mu_{(v_m \ominus D_{0i}) \cap M_{ij} \otimes (t_m \ominus t^i)}(u)\}\} \quad (4)$$

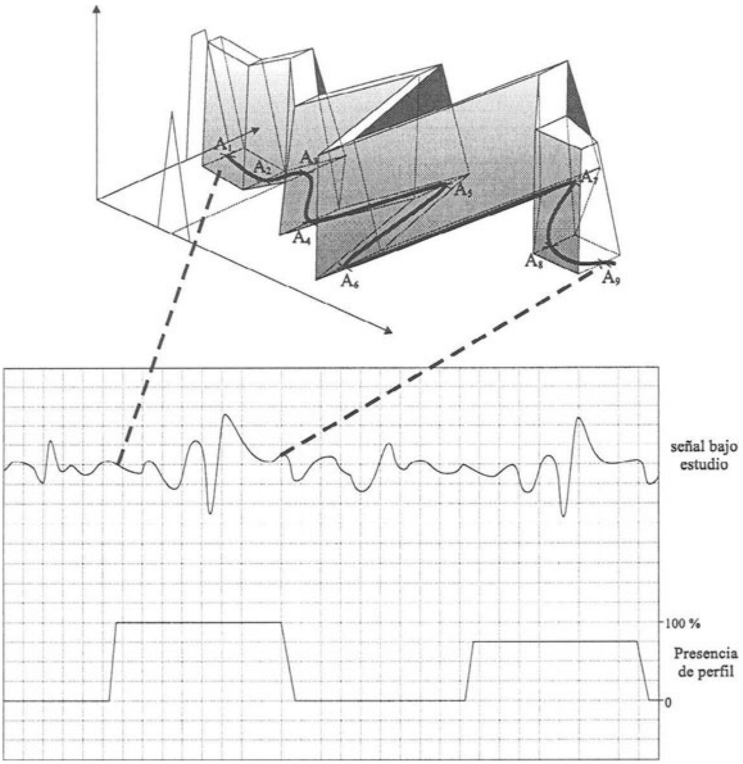


Fig. 3. Location of a profile the morphology of which is outlined in figure 1. The guideline section of the matching process is the R_{67}, written as *' a slope rising sharply from a low value.* This is the most significant section, and as such, the most adequate reference with which to initiate the profile location process.

Up until now we have located a section on the signal, but what we are really interested in is obtaining some of its extreme values. In this way we will have in turn located the FTP, and we will be able to apply the assignment procedure described in the previous section, but with a different assignment order. The procedure for the obtention of the initial moment of the section is realised in a very simple manner, by applying expression 4 to the initial moment $\mu_{ij}^{FTP}(p_n = (v_n, t_n = t^i))$ $\forall p_n \in P$. Thus we are changing the meaning of this expression, obtaining the degree of membership of each sample to a section of the slope M_{ij}, started at D_{0i}, of a duration L_{ij} and which begins in this sample. In other terms, this expression corresponds to the degree of possibility of each signal sample by being the beginning of a section of the aforementioned characteristics, and with which by instancing the constraints of the section on the signal we can refine the constraint on its beginning $\pi_{0i}^{L'}(t_n) = \mu_{ij}^{FTP}(p_n = (v_n, t_n = t^i))$, $\forall p_n \in P$.

4 Discussion

In his study Steinmann [7] deals certain aspects related with modulating fuzzy linear sections for the identification of trends on the signal, demonstrating the great adaptability of a fuzzy-set-based representation for the analysis of signals in real domains.

We believe that our proposal is much more ambitious, since Steinmann tackles the matching of one single section over a precise time section. In our approach the section is part of a more complex profile, in which the extension of each one of the sections that comprise the profile is delimited in a fuzzy manner. To this one has to add the capabilities, both expressive as well as analytical, of working on constraint networks, a fundamental basis of our model. In this way, we are developing a profile representation language which affords the expert a easily understood description of profiles in terms which are close to natural language.

Within the framework of the real applications of the FTP model, we are currently working on its integration, at a representational and reasoning level, into a perception system intensive coronary care units (ICCU) based on blackboard architecture [6].

References

1. R. E. Bellman and L. A. Zadeh. Decision-making in a fuzzy environment. *Management Science*, 17(4):141–154, 1970.
2. D. Dubois, H. Fargier, and H. Prade. Possibility theory in constraint satisfaction problems: handling priority, preference and uncertainty. *Applied Intelligence*, 6:287–309, 1996.
3. H. Fargier. *Problemes de satisfaction de constraintes flexibles, application a l'ordonnancement de production.* PhD thesis, Universite P. Sabatier, Toulouse, France, 1994. In French.
4. P. Félix, S. Barro, R. Marín, M. J. Taboada, and A. Engel. Patrones temporales borrosos en la supervisión de pacientes. In *Actas del V Congreso Español sobre Tecnologías y Lógica Fuzzy*, 1995. In Spanish.
5. P. Félix, S. Fraga, R. Marin, and S. Barro. Trend detection based on a fuzzy temporal profile model. In *EIS'98*, 1998. Accepted.
6. S. Fraga, P. Félix, M. Lama, E. Sánchez, and S. Barro. A proposal for a real time signal perception specialist. In *EIS'98*, 1998.
7. F. Steimann. The interpretation of time-varying data with diamon-1. *Artificial Intelligence in Medicine*, 8:343–357, 1996.
8. L. A. Zadeh. The concept of a linguistic variable and its application to approximate reasoning. *Information Science*, 8:199–249, 1975. Part 1.

Performance of a Smart Velocity Sensor: The Impulse Retina

MARILLY Emmanuel[1], COROYER Christophe[2], CACHARD Olga[1],
FAURE Alain[1]

[1]Laboratoire LACOS, Université du Havre
Place Robert Schuman, 76610 Le Havre, France
email : marilly@iut.univ-lehavre.fr
[2]ESIGETEL, 1 rue Port de Valvins
77215 Avon-Fontainebleau cedex, France

Abstract. In the context of mobile robot, we have developed an artificial neural network permitting a pre-processing of foveal vision : the Retina model. This model is adaptative and its multi-resolution allows to detect a large scale of velocities. The aim of this study is to use the Retina to detect the motion and extract the velocity vector of a time sequence image. From impulse output signals of Retina we extract the pertinent parameters which encode the motion by time frequency analysis.

1 Introduction

There are several approaches permitting the motion detection and the motion estimation of moving body in a scene. A method consists in processing a time sequence image [1]. The main methods are based respectively on the differential approaches (or gradient method [2]), the critical feature matching [3] and the spatio-temporal filtering [4] [5].

The different properties associated to each of these methods are optimized for experimental conditions well-defined [6]. The gradient methods depend on estimating first and second order derivatives of pixel intensity, this causing image noise and spectrum problems. The correspondence methods are widely implemented but the implementors often resort to great simplifications and they limit the constraints on the system operation. The correspondence method can be beset with combinatorial explosions of matching possibilities between two time sequence images. These inconveniences, have motivated research into spatio-temporal filtering methods. The spatio-temporal methods are linked to the biological vision knowledge [7]. Taking as model the biological structures that implement the motion analysis may develop artificial systems that tend to reach the power of nature.

In order to reproduce the mechanism of vertebrate vision, we have evolved a first simplified model of foveal vision : the Retina model. It's based on an artificial neural network organized to simulate the radial variation of the visual acuity. The implement of analogic and impulse neurons enabled us to encode the information. The main interest of this model is its response to the stationary stimuli or in motion, that can be distinguished according the shapes and the velocities. This model is adaptative and its multi-resolution allows to detect a large scale of velocities (low velocities (fovea), high velocities (peripheries)).

An adapted analysis and representation of retinal signals enabled us to extract the pertinent information given by our sensor, allowing us to detect the motion and extract the velocity as well as to do pattern recognition.

2 The Impulse Retina Model [8]

2.1 Retinal Processing of the image [9]

In the biological retina, the visual acuity decreases from the center of the retina (fovea) to the periphery. This property is attributed to the receptor density variation which decreases from the center to the periphery. To simulate this non-uniform distribution of the retinal receptor we recombine the image according to the principle of the multi-resolution pyramid [10] by superimposing concentric circular area. Each area has different resolution levels. The central point (i_0,j_0) is the centre of 5 concentric rings of different resolutions (l_0 is the initial resolution) whose radii are (1):

$$R_k(l_0) = 3 * 2^{(l_0-1+k)} \text{ with k=\{0,..,5\} and } l_0 \in (1,2,3,4,5). \qquad \textbf{(1)}$$

To represent the image I={X_{ij}} where X_{ij} is the grey level of the pixel at the resolution level h, we use the recurrent computation of the gaussian convolution at each point.

We obtain (2) :

$$X_{ij}^{h} = \sum_{p,q} W(p,q).X^{(h-1)}_{\left(i-2^{h-1}.p,j-2^{h-1}.q\right)} \qquad \textbf{(2)}$$

where W is the gaussian kernel or weight function with p,q∈(-2,-1,0,1,2).

The value so obtained, are put in the input layer of the neural network. As it will be shown below, this organizes the receptive field of neurons, and filters the different spatial characteristics. The figure 1 sums up the functional diagram of the Retina .

2.2 The Retinal neural network [11]

This network is comparable to a simplified biological retina model constituted by bipolar cells (Xe and Xi) and ganglion cells (Y). The neural network has two input layers (Xe Excitatory neuron layer, Xi Inhibitory neuron layer) and one output layer (Y Impulse neuron layer). The aim of the feed-back in the model is to ameliorate the adaptivity of the Retina (adaptation to the different pixel intensity).

The behaviour of the excitatory neuron layer and inhibitory neuron layer is described by first order differential equations similar to the Grossberg model [12].

The behaviour of impulse neurons (similar to the biological neurons) is computed in considering the Hodgkin-Huxley's formalism for the dynamics of the neuron's membrane potential [13].

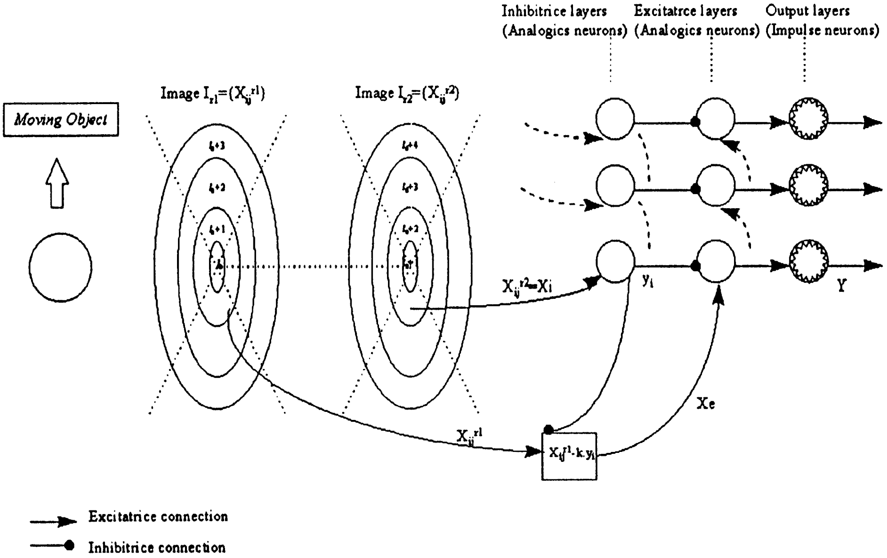
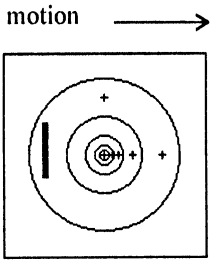

Inhibitrice layers (Analogics neurons)
Excitatrce layers (Analogics neurons)
Output layers (Impulse neurons)
Moving Object
Image $I_{r1}=(X_{ij}^{r1})$
Image $I_{r2}=(X_{ij}^{r2})$
t_0+3
t_0+2
t_0+1
t_0+4
t_0+3
t_0+2
$X_{ij}^{r2}=X_i$
y_i
Y
Xe
X_{ij}^{r1}
$X_{ij}^{r1}-k.y_i$
Excitatrice connection
Inhibitrice connection

Fig. 1. Scheme of the Retina

3 Simulations and Comments

3.1 First Simulations

To study the motion sensibility of this artificial retina we compute the neurons response by using numerical sequential algorithm. The simulations are done with moving stimuli of various shapes. For this first simulations, three different series of stimuli are investigated : squares, stripes moving in direction of their long axis, and stripes moving perpendicular to the directions of their long axis. The stimuli have constant intensity (250). The background is also constant (0). All the stimuli have been moved at constant velocity V. The stimuli velocity is defined by the ratio of pixels number moved (n) on the time unit passed (h).



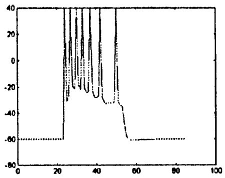


Fig. 2. A stimulus moving on the Retina **Fig. 3.** Example of fovea neuron's response

The signals given by the numeric model have a complex structure linked to the stimuli characteristics. It's interesting to analyse this signal in order to extract the pertinent elements attached to this spatio-temporal character. To do this analysis we have used the Wigner Ville transformation that allows a time frequency representation of the signal.

3.2 Time frequency analysis

3.2.1 Definition

Wigner Ville representation is the most used time-frequency representation, for non stationary signals, because of its simple implementation and its quite good results [14]. The Wigner Ville transformation is defined by (3)

$$PWV(t,\nu) = \int_{-\infty}^{+\infty} h(\tau) x\left(t+\frac{\tau}{2}\right).x^{*}\left(t-\frac{\tau}{2}\right).e^{-j2\pi\nu\tau} d\tau \qquad (3)$$

where x(t) : temporal signal, and h : observation window. We estimate the Wigner Ville representation by (4):

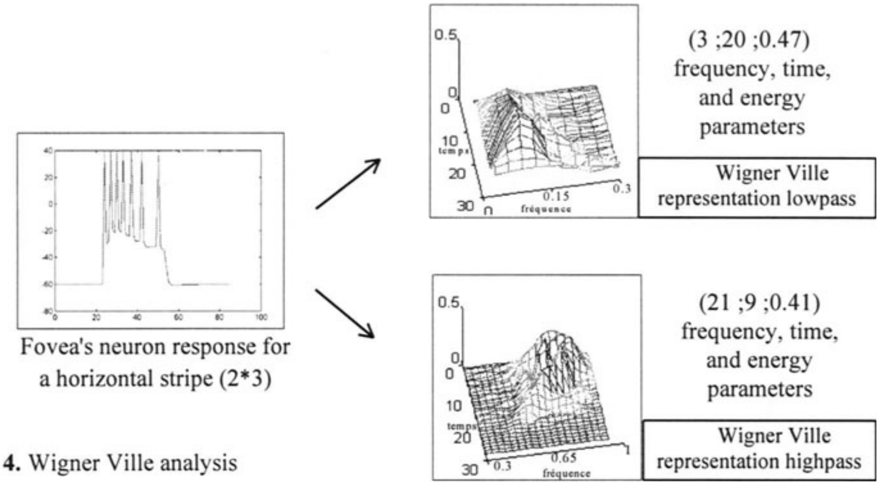
$$E[WV_x(n,f)] = \frac{1}{N} DFT\left(\sum_{k}\sum_{m} F[m-n,2k].z[m+k].z^{*}[m-k]\right) \qquad (4)$$

where z[n] is the Hilbert transform of x[n] and F[i,j] is a weighting function. The analytic signal z[n] is processed to cancel interference due to negative frequencies terms.

3.2.2 First Results

The impulse signal coming from the Retina has two components (a low frequency and a high frequency). Then, we decided to separate the signal into two parts, by filtering with a lowpass and with a highpass filter. Thus we will obtain two Wigner Ville representations without interference. We extract from each representation the vector of amplitude maxima. The vector is composed of frequency, time (ms), and amplitude parameters.

An adapted representation of those parameters allows to visualise the properties and the particularities of each component.



Fovea's neuron response for a horizontal stripe (2*3)

Fig. 4. Wigner Ville analysis

4 Generalization

We select about thirty neurons in each level of resolution of the Retina. The chosen neurons have characteristic positions so that their receptive field overlaps all the surface of the Retina model. Of those neurons we extract the vectors composed of frequency, time and energy localisation. By applying these computation process and representing those parameters according to the position of the neurons on the retina, we will have a temporal, frequency and time representation highpass and lowpass.

For a horizontal stripe (2x3) moving in direction of their long axis, we obtain the following results (Fig5,6 and 7).

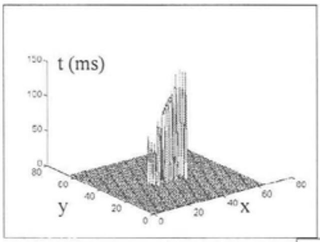


Fig.5. highpass time representation (x,y,t)

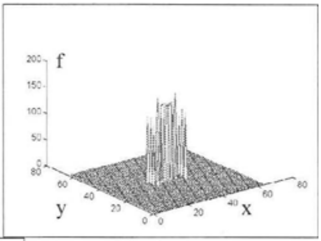


Fig.6. highpass frequency representation (x,y,f)

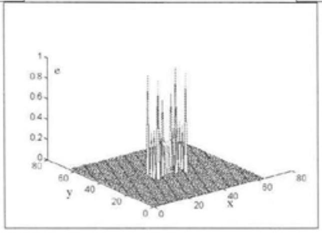


Fig.7. highpass energy representation (x,y,e)

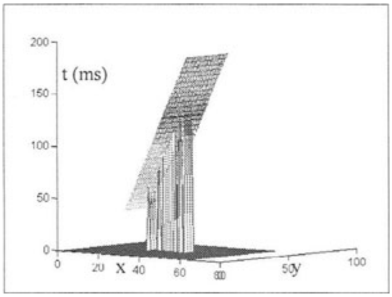
The temporal representations (lowpass and highpass (Fig 5)) are the same. They are independent of the form. We have a motion representation in x-y-t space. We are in the same case defined by Adelson and Bergen [4] in which x and y are the two spatial dimensions and t is the temporal dimension. The three-dimensional spatiotemporal diagram is shown in figure 5. The slant of the cross section (Fig5) reflects the velocity of the motion. We will be able to detect and define the global direction of the motion.

The frequency representations (x,y,f) are very interesting (Fig 6), but they aren't linked to the motion stimuli, they define the spatial frequency of the stimuli. This signature depends exclusively of the global shape of the stimuli, then we used this parameters to do the pattern recognition. The shape characteristics appear in this representations (similar to symmetry...)

The representations (Fig7) of the spatial distribution of the energy (x,y,e) will also be used to do the pattern recognition.

5 Velocity Estimation

The set of values of the temporal representation (Fig 5) define approximately a plan (variable according to the motion type). A linear regression will enable us to determine the plan equation $t = ax + by + c$ which associate the time t to the position (x,y). Since the slant reflects the velocity of the motion we will be able to extract the vector $\vec{V} = \left(v_1, v_2\right)$ which defines the velocity of the stimulus on the Retina [5].



$$\xrightarrow[\text{Velocity vector}]{} \quad \vec{V} = \begin{pmatrix} 0.2 \\ 0 \end{pmatrix}$$

Fig.8. time representation + plan

This first simulations with shapes of velocity and direction that are well-known, allow us to study the behaviour of the Retina. We find that the vector $\vec{V}$ is independent of the shape of the stimuli. Experimentally we find the velocities vector value defined theoretically. For instance in figure 8, the stimulus moves following the axis x at the theoretic velocity of 1 pixel all the 0.2 sec. Experimentally the process allows to find a velocity vector of Vx= 5 pixel/sec and Vy=0.

The table 1 shows different results obtained by the Retina model. These results are characteristic, because they permit to verify the sensor behaviour.

Stimulus	square 3x3	square 3x3	square 5x5	h.s. 2x3	h.s. 2x3
Theoretical velocity vector	$\vec{V}\begin{pmatrix}0.4\\0\end{pmatrix}$	$\vec{V}\begin{pmatrix}0.2\\0\end{pmatrix}$	$\vec{V}\begin{pmatrix}0.2\\0\end{pmatrix}$	$\vec{V}\begin{pmatrix}0.4\\0.4\end{pmatrix}$	$\vec{V}\begin{pmatrix}-0.4\\-0.4\end{pmatrix}$
Experimental velocity vector	$\vec{V}\begin{pmatrix}0.42\\0\end{pmatrix}$	$\vec{V}\begin{pmatrix}0.2\\0\end{pmatrix}$	$\vec{V}\begin{pmatrix}0.2\\0\end{pmatrix}$	$\vec{V}\begin{pmatrix}0.41\\0.39\end{pmatrix}$	$\vec{V}\begin{pmatrix}-0.4\\-0.41\end{pmatrix}$

Table 1. Comparison between Experimental and theoretical results (h.s. : horizontal stripe).

The very little difference between the theoretical and experimentical results allow us to conclude that we can perfectly detect all the directions and the Retina multiresolution enables to detect a large scale of velocity.

6 Application to a Sequence of Images

Let's valid this model by using a real time sequence image. The adaptivity and the multiresolution of the Retina take both all their importance when we use real images. In the sequence "taxi of Hamburg" (Fig 9), the Retina is used to detect the motion of the three cars.

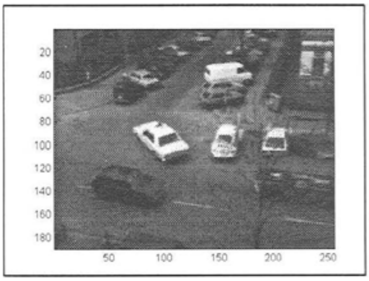


Fig.9. Time sequence image

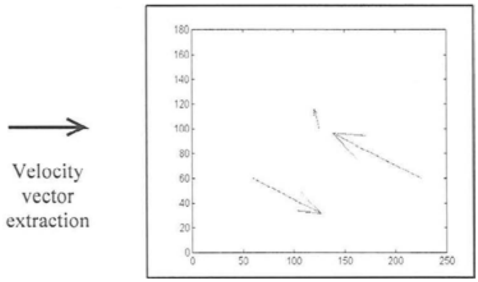


Fig.10. velocity vectors representation

The procedure used to extract the velocity vectors of the three moving car is quite simple. We follow the method developed by Y. Zeevi [15] for foveating sensor. A coarse Retina (with low resolution scene coverage) is used to detect the area of interest. Then a fine Retina (with high resolution) is created in each interesting area and we use the previous describe method to extract the velocities vectors.

The velocity vector of the three moving vehicles in the image sequence are shown figure 10. The adaptivity of the Retina was fundamental to extract from this time sequence image the velocities vector of the three cars (Fig 10) in spite of the perturbations (noise, different intensities level...).

7 Conclusions

The utilization of a sensor of biological inspiration: that is the Retina model in order to detect motion and extract the velocity vector, brings many advantages. First of all it uses the foveal vision that allows to detect a large scale of velocities. It's a very robust sensor (many neurons). The noise problem is solved by using the multi-resolution image pyramid. The Retina is adaptative (feed-back) and the time processing is relatively short. We will bring some modifications to the Retina (for example using the log-polar transform) to allow the detection of complex motion like rotation, expansion. We should ameliorate the time of process by modifying our algorithms to do real time processing. We shouldn't forget to use the frequency representation to do the pattern recognition. The first results of classification of the frequency parameters with neural network (back-propagation) are very satisfaying. For three groups of stimuli the neural networks give 100 % of correct classification. The final aim is to implement this sensor on a mobile robot (obstacle avoidance, supervision, trajectory prediction).

References

1. Laplante et Stoyenko, "real-time imaging, theory, technique et applications" IEEE Press 1996.
2. Horn et Schunck, "determining optical flow" Artificial intelligence, vol 17, pp 185-203, 1981.
3. Singh, "incremental estimation of image-flow using a kalman Filter", IEEE wokshop on visual motion (princeton, nj) pp36-43, 7-9 October 1992.
4. Adelson et Bergen, "spatiotemporal energy models for the perception of motion" J. Optical Society of America, vol A2, no 2 pp 284-299, 1985.
5. Watson et Ahumada, "Model of human visual-motion sensing" J. Optical Society of America, vol A2, no 2 pp 322-342, 1985.

6 A. Singh et P.K. Allen, "A real time hierarchical model for optic flow determination via spatiotemporal frequency channels" in Proc. DARPA image understanding workshop (Washington, dc), pp 961-967, april 1988.

7. Grzywacz et Yuille "A model for the estimates of local image velocity by cells in the visual cortex" Proc. Royal Soc. London (B), vol 239, pp129-161, 1990.
8. Cachard olga 1996 'Modélisation de la vision fovéale des vertébrés : application à la detection des vitesses et des tailles des objets' Thése de doctorat de l'université du havre 1996.
9. Faure, Rybak, Cachard 1994, 'RETINA: a Model of visual information processing in the retina neural network' ICANN'94 the European conference on artificial Neural Networks.1994.
10. Burt, 1988, 'Smart sensing within a pyramid vision machine' Proc. of the IEEE, vol 76, pp1006-1015, 1988.
11. Marilly, Cachard, Rybak, Gusakova, Faure 1995, 'Impulse Retina model : comparison between biological data and the results of the simulation' second international symposium on neuro informatics and neuro computers, rostov on don, Russia, September 1995.

12. Grossberg : 'Non linear neural networks : principles, mechanisms, and architecture' 1988 voll p17-61 neural networks.

13. Hodgkins et Huxley, 1952, ‘A quantitative description of membrane current and its application to conduction and excitation in nerve’, Journal of physiology 117, pp500-544.

14. Claasen 'The Wigner Distribution a tool for time-frequency signal analysis' parts I II and III, T.A.C..M. Claasen and Mecklenbäuker, Philips journal of research, vol 35 n 3,4-5,6, 1980.

15. Y. Zeevi, R Ginosar 'Neural Computers for Foveating Vision Systems' in Advanced Neural Computers. Eckmiller (editor). Elsevier Science Publishers 1990 p323-330.

Two Methods of Linear Correlation Search for a Knowledge Based Supervised Classification

Amel Borgi[1,2], Jean-Michel Bazin[2], Herman Akdag[1,2]

[1] LIP6, Université P. et M. Curie, 4, Place Jussieu, Boîte 169,
75005 Paris, France
{Amel.Borgi, Herman.Akdag}@lip6.fr
[2] LERI, Université de Reims, Rue des Crayères, B.P. 1035
51687 Reims Cedex 2, France
jm.bazin@univ-reims.fr

Abstract. We present an image classification system based on a supervised learning method. The learning phase consists in an automatic rules construction: « knowledge acquisition » from training pixels is automatic. The obtained rules are classification ones : their conclusions are hypotheses about the membership in a given class. An inference engine uses these rules to classify new pixels.

The building of the premises of the production rules is realized by linear correlation research among the training set elements. In this paper, we present and compare two methods of linear correlation searches : the first is done among all the training set without distinction of classes, and the second is an intra-classes search. An application to image processing in the medical field is presented and some experimental results obtained in the case of medical human thigh section are reported.

1 Introduction

We are interested in the classification of medical images and we work upon classification by automatic rules generation under supervised learning.

In every learning system, two points are of particular interest :

- knowledge representation and the know-how that are embedded in the expert system's rules,
- an inference engine that allows the system to reason about this knowledge.

The first aspect is easy to realize in some classical domains in which fragments of knowledge are well structured. However, this work is very delicate in the pattern recognition domain in which expert's rules will allow to determine the belonging of a pixel to a class. For instance, a pixel will be classified as bone, muscle, ...

In this domain, it is practically impossible to obtain from an image classification given by a medical specialist a coherent and complete set of rules. The only trusted informations he can give us, are relations between pixels and classes such as $P_i \in C_j$, where P_i is a pixel and C_j is a class defined by the domain specialist. This method limits mistakes during knowledge acquisition. Therefore, starting from the expertise given by a medical expert such as $P_i \in C_j$, our system builds rules that allow it to classify new pixels.

One of the particular interest of our work consists in this base of rules automatically constructed [1]. To achieve this goal, the picture processing expert represents a pixel by a vector : each pixel feature corresponds to a vector component. Thus we have a classified vectors set, the training set, that we use to generate the base of rules. The building of the premises of the rules is realized by linear correlation search among the components of the training set vectors. The originality of our work consists in proposing two methods of linear correlation searches. The first one consists in a linear correlation search among all the training set elements, without any distinction. The second type is an intra-classes linear correlation search: the training set vectors are gathered together according to their class, and a linear correlation search is done among the components of the vectors of each class considered separately. The second method allows to characterize each class, and to gather discriminate informations for a further classification. We claim that this method is more performant for the classification than the first one.

Another important contribution in the rule building is the discretization of pixel components. Following Vernazza [1] and Ginsberg [2], we introduce a multi-valued representation of knowledge.

As in most applications, picture recognition is concerned with uncertainty [3] [4]. In fact, the conclusion parts of the rules are hypotheses about membership in a given class. Each hypothesis is accompanied by a belief degree that can be a probability [5] or a certainty factor [6] [7].

The generated rules base is used by an inference engine to classify new pixels. The rules fired are those of which premises match the components of the new pixel to classify. Uncertainty management is done by computations on the belief degrees of the fired rules. These computations allow to choose a final class for the new pixel.

The purpose of this paper is to compare the two methods of linear correlation search. In section 2, generalities are presented about the rules construction. Then, in section 3, we describe the rules construction according to a linear correlation search with all the classes regrouped. Specificities of linear correlation intra-classes are presented in section 4. Section 5 covers the inference system and the classification techniques. Discussion and performance results are provided in section 6. Finally, section 7 concludes the study.

2 Generalities on the Rules Construction

2.1 The Training Set

The treated data are medical image data and they correspond to a human thigh cryo-section. By considering the digital picture of such anatomic section, the medical expert classifies different zones. Every pixel of the picture is classified into : bone pixel (class C_1), muscle pixel (class C_2), marrow pixel (class C_3), fat pixel (class C_4). Pixels are represented by vectors, each pixel feature corresponds to a vector component. A n components vector $(X_1, X_2, ..., X_n)$ is associated to every pixel. In the set of classified pixels, we can now choose : N pixels of the class C_i, $i \in [1,4]$.This set of 4×N pixels constitutes the training set, from which we will build the rules.

2.2 Morphology of the Rules

Let $C_1, ..., C_c$ be the possible classes defined by the medical expert. Let $X_1, ..., X_n$ be the components of the vector representing the pixel features. The generated rules are of the type : $\mathbf{E_1, E_2, ..., E_m \rightarrow C, d}$
where E_i is a condition of the form : X_j is in [a,b], X_j is the j^{th} vector component and the interval [a,b] is issued from the discretization technique described in [1]. **C** is a hypothesis about the membership in a class with a belief degree **d**. In this paper, we propose to represent this degree by a classical probability [5] [8].
Classification techniques vary greatly according to the problem. Methods using the Certainty Factors (CF) theory are usually used to manage uncertainty in based-rules systems [6] [7]. In our case, the CF approach will allow us to build the rules, and then, in further works, to represent uncertainty. The rules construction needs to specify the following points :

- how to build the premise of each rule,
- how to choose the interval [a,b],
- how to determine the value of the belief degree.

The two methods of linear correlation search we propose, give an answer to the first question : they allow us to build the premises of the rules.

3 Building the Rules: a Linear Correlation Search upon all Classes

The first method proposed to build the premises of the rules is based on a linear correlation search among the components of all the training set vectors, without any distinction [9].
To determine which components are regrouped in the different rules, we first use the correlation matrix $R=(r_{i,j})_{n*n}$, where $r_{i,j}$ is the coefficient of linear correlation between the components X_i and X_j of the training set vectors.

$$R = \begin{bmatrix} 1 & r_{1,2} & \cdots & r_{1,n} \\ r_{2,1} & 1 & \cdots & \\ \cdots & & & \\ r_{1,n} & \cdots & & 1 \end{bmatrix}$$

Then to decide which components are correlated, we fix a threshold θ, and we consider that (X_i) and (X_j) are independent when $|r_{ij}|<\theta$.

According to the CF approach [6][7], we decide to group in the same premises, all components that are correlated. For example, in the case of pixels represented by 5-components vectors, if the thresholded correlation matrix obtained is :

$$\mathrm{Rt} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

we will then group together (X_1,X_2,X_3) and (X_4, X_5). Thus, we obtain a partition of the components set $\{X_1,X_2,X_3,X_4,X_5\}$.

The next point is to decide how to choose the interval [a,b]. This point deals with the discretization technique. We decompose the global range of a component into a finite number M of subranges of equal width (M is the subdivision cardinal).

Premises of rules are obtained by considering, for each correlated components subset, an interval for each component in all possible combinations. For example with 5 components regrouped into the 2 subsets $\{X_1,X_2,X_3\}$ and $\{X_4,X_5\}$, and with 3 intervals for each component, we obtain $3^3+3^2=36$ combinations. Thus, in our example, if we divide the global ranges I_j of each component X_j into three subranges of equal width, denoted rg_0, rg_1 and rg_2, we can build the following groups of premises :

if (X_1 in rg_a) and (X_2 in rg_b) and (X_3 in rg_c)   with a,b,c in {0,1,2}
if (X_4 in rg_d) and (X_5 in rg_e)   with d,e in {0,1,2}

Each premise thus obtained leads to c rules, where c is the total number of the classes. So to compute the number of possible rules, we multiply the number of premises by the number of classes (in our example, c=4, and we obtain 36×4 rules).

The final step of the rule generator is to compute the belief degree associated to each conclusion. It is represented by a classical probability [5] [8]. The estimation of each rule probability is computed using the well-known Bayesian formulas on the whole training set.

Finally, we obtain rules like : $E \rightarrow C$, P(C,E) where E is the conjunction of events of the form E_1, E_2, ...,E_m already mentioned above, C a hypothesis about the membership in a given class and P(C,E) the conditional probability of this class.

Fig. 1 resumes the rules construction operations in the case of a linear correlation search upon all classes.

The number of generated rules may be important. This is the reason why we developed an optimization module that allows to reduce the number of rules. This module, used in both correlation search methods, eliminates the neutral rules which are those having a zero belief degree.

Let's now present the second correlation search method that is the intra-classes one.

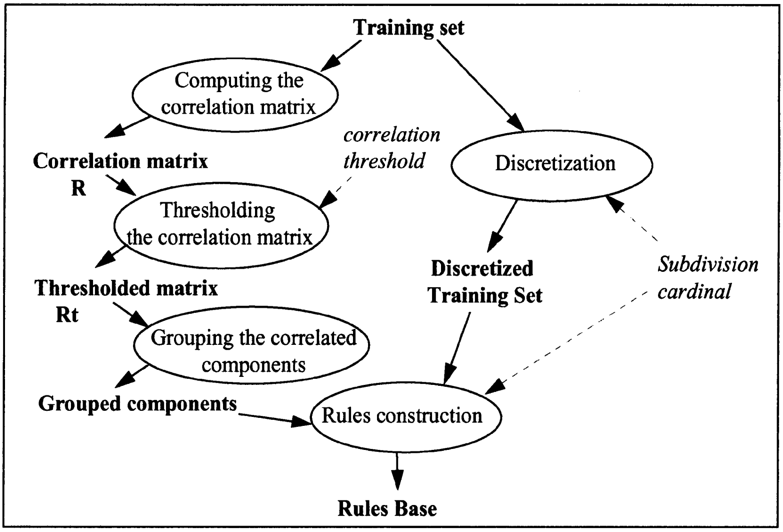


Fig. 1. Linear correlation search upon all classes

4 Building the Rules: an Intra-classes Linear Correlation Search

This method differs from the one described above concerning the construction of the rules premises, and more precisely the combinations choices of components brought together. In this case, the linear correlation search is done among the components of the training set vectors belonging to the same class. For this, we need a first treatment to cut up the training set and thus pixels of the same class are gathered together. The subsets obtained are denoted TS_1, ..., TS_k, ... TS_c, where c is the total number of classes. TS_k contains all the training set pixels of the class C_k. In our example, we will have four sub-sets :

- TS_1 : bone pixels (class C_1)
- TS_2 : muscle pixels (class C_2)
- TS_3 : marrow pixels (class C_3)
- TS_4 : fat pixels (class C_4)

For each subset TS_k, we compute a correlation matrix $R^k=(r^k_{i,j})_{n*n}$. We denote by $r^k_{i,j}$ the coefficient of linear correlation between the components X_i and X_j of the TS_k vectors.

$$R^k = \begin{bmatrix} 1 & \overset{k}{r}_{1,2} & \cdots & \overset{k}{r}_{1,n} \\ \overset{k}{r}_{2,1} & 1 & \cdots & \\ \cdots & & & \\ \overset{k}{r}_{1,n} \cdots & & \cdots & 1 \end{bmatrix} \qquad \text{where} \quad k \in \{1,2,\ldots,c\}$$

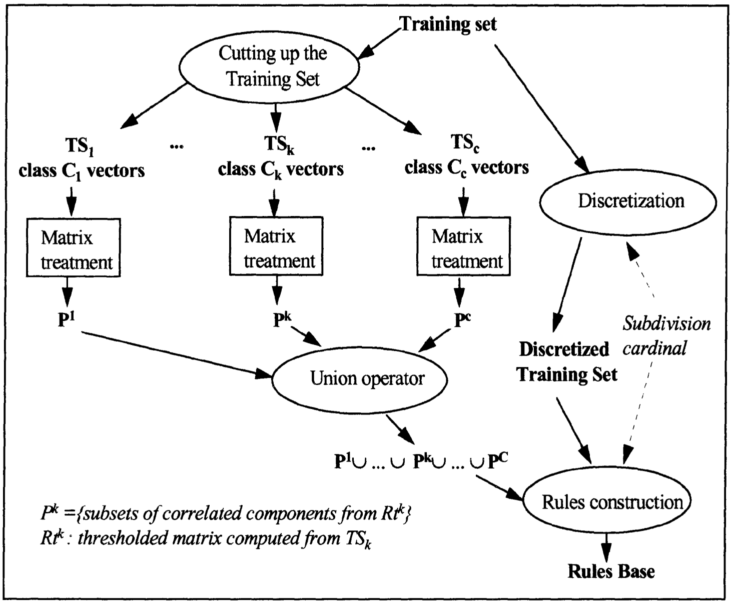


Fig. 2. Intra-classes linear correlation search. The matrix treatment is the same as in the linear correlation search upon all classes (Fig. 1). It consists in computing the correlation matrix R^k from the training set TS_k, then in thresholding R^k to obtain a thresholded matrix Rt^k used to determine the correlated components (P^k).

To determine which combinations of components should be brought together in the rules premises, we proceed as we did in the first method. Indeed, to decide which components are correlated, we fix a threshold θ, and we consider that (X_i) and (X_j) are independent if $|r_{ij}|<\theta$, the same threshold being used for all the matrixes. For each thresholded matrix Rt^k obtained from R^k, we have subsets of correlated components, these subsets form a partition P^k of the components set $\{X_1, X_2, ..., X_n\}$. All the subsets, obtained from the different matrixes, are regrouped by the union operator to obtain a final set of correlated components subsets, this final set is not necessarily a partition of $\{X_1, X_2, ..., X_n\}$. Then, we group in the same premise, the components that are correlated.

In our example, 5-components vectors and 4 classes lead to 4 matrixes (5×5). For instance, we may obtain the following thresholded correlation matrixes:

$$Rt^1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \quad Rt^2 = Rt^3 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix} \quad Rt^4 = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

So, the correlated components are : for Rt^1: (X_1), (X_2), and (X_3, X_4, X_5),
for Rt^2 as for Rt^3: (X_1, X_2, X_3) and (X_4, X_5),
for Rt^4 : (X_1, X_2), (X_3) and (X_4, X_5).

The final set obtained through the union of all these subsets is :

$$\{\{X_1\}, \{X_2\}, \{X_3, X_4, X_5\}, \{X_1, X_2, X_3\}, \{X_4, X_5\}, \{X_1, X_2\}, \{X_3\}\}$$

The correlated components will be brought together in the premises of the rules. Then to end the premises construction, we proceed exactly like in the first correlation method : we decompose the global range of a component into a finite number M of subranges of equal width, and the premises of the rules are obtained by considering for each correlated components subset, an interval for each component in all possible combinations.

Each premise obtained in this way leads to c rules, where c is the total number of classes. The final step of the generator system is then to compute the belief degree associated to each conclusion. We remind that the belief degrees are represented by classical probabilities [5] [8]. They are computed as they were in the first method: that is to say by using the Bayesian formulas on the whole training set without any distinction of classes. The obtained rules have the same morphology as those generated with the correlation search upon all classes. Fig. 2 shows the main steps of this rules construction.

5 The Inference Engine

Whatever the construction method was, the rules were generated for the purpose of a further classification use. In fact, the inference engine will exploit the rules base to classify new pixels [10]. The main difference between our system and more classical ones lies in the treatment of uncertainty.

For a new pixel V to classify, the system begins by computing its discretized components. This is done according to the discretization technique used in the building of the rules premises. Then the inference engine fires the rules which the new pixel discretized components match. In the next step, the system regroups the fired rules according to the class of their conclusion parts. In other words, for every class C_i we obtain a set of rules, denoted Rules(C_i), containing the fired rules that conclude to the class C_i. We then have to compute a final belief degree associated to each class, for this we use a co-norm S such as :

$$S(p,q) = \max(p,q) \quad \text{or} \quad S(p,q) = p+q-p*q$$

We remind that a co-norm is a function used in knowledge uncertainty treatment. It constitutes an example of aggregation function [11] [12]. A co-norm is a real function S : $S : [0,1]x[0,1] \rightarrow [0,1]$

having the following properties :

c1 : $S(1,1)=1$

c2 : $S(p,0)=S(0,p)=p$

c3 : $S(p,q)=S(q,p)$

c4 : $S(p,q)\leq S(r,c)$ if $p\leq r$ and $q\leq c$

c5 : $S(p,S(q,r))=S(S(p,q),r)$

Finally, the winner class associated to the new pixel is the class where the final belief degree is maximum.

6 Results and Discussions

The learning phase or automatic rules construction is done by the *generator system* and the classification phase ensured by the inference engine is done by the *inference system*. Conceptually, both systems (*generator* and *inference* systems) can be considered as a whole package S of which inputs are the training set of classified pixels, the new pixels to classify and the 4 following parameters:

- the correlation threshold (θ),
- the subdivision cardinal : number of subranges issued from the discretization (M),
- the linear correlation search method (upon all classes or intra-classes),
- the computation method of final degrees (co-norm).

S allows the classification of new pixels. It was implemented in the C language and developed modularly : the *generator* and the *inference* systems are two independent modules.

For each combination of parameters and for a given training set, the final rate per cent of successful classification is obtained by the application of the following method.

6.1 Cross-reference Method

Let E be a set of pixels classified by a medical expert. E is splitted into 4 equal range parts E_1, E_2, E_3, E_4. Starting from the set $E_1 \cup E_2 \cup E_3 = E_{123}$, the generator system builds a set of rules R_{123}.Then, R_{123} is used to classify the pixels of E_4, i.e. E_{123} is used in training and E_4 in testing. When comparing the classification obtained by applying R_{123} to E_4 to the classification of pixels of E_4 given by the medical expert, we can compute the rate per cent T_{123} of successful classification.

Applying the same method by circular permutation on the sets E_1, E_2, E_3, E_4, we obtain four rates per cent of classification. The final rate per cent given in output of S, and which is the classification rate, is the average of these four rates.

6.2 Main Results

Different parameters combinations were tested. We fixed the parameters values in an empirical way to find the best combination :

- correlation threshold (θ) : 0.5, 0.8, 0.9, 0.95
- discretization step (M) : 3, 7
- linear correlation search method : upon all classes, intra-classes
- computation method of final degrees, two co-norms were tested : max, $p+q-p*q$.

Giving to each parameter all these values, we obtain 32 combinations. We list below the results obtained in the case of human thigh image data (Tables 1. and 2.). In this case, pixels are represented by 5 components vectors, and the number of classes is 4 (bone, muscle, marrow and fat). The figures under 'ratio' present results for the intra-classes method divided by the corresponding figure for the correlation search upon all classes.

We can immediately note that in most cases the results obtained with the intra-classes correlation search are better than those obtained with the correlation search upon all classes. The improvement rises to 23% for a step value of 7, a threshold value of 0.95 and with the co-norm max. The only case where the intra-classes method leads to similar results, and more precisely 1% worse than those obtained with the first method, is the case where the classification rates are very good (about 96%). This case corresponds to a step of 7 and a threshold of 0.5.

Table 1. Results with linear correlation search upon all classes

Subdivision cardinal :		3		7	
Threshold	co-norm :	max	p+q-pq	max	p+q-pq
0.95	Nb of optimised rules	34	34	54	54
	Classification rates (%)	**79,00**	**83,42**	**77,08**	**87,17**
0.90	Nb of optimised rules	29	29	57	57
	Classification rates (%)	**79,17**	**83,58**	**78,33**	**87,75**
0.80	Nb of optimised rules	26	26	62	62
	Classification rates (%)	**79,42**	**83,83**	**80,33**	**88,67**
0.5	Nb of optimised rules	22	22	58	58
	Classification rates (%)	**86,83**	**86,83**	**96,42**	**96,75**

Table 2. Results with linear correlation search intra-classes

Subdivision cardinal :		3				7			
Threshold	co-norm :	max		p+q-pq		max		p+q-pq	
		ratio		ratio		ratio		ratio	
0.95	Nb of optimised rules	60		60		138		138	
	Classification rates %	**88,17**	**1,12**	**88,00**	**1,05**	**94,58**	**1,23**	**95,75**	**1,10**
0.90	Nb of optimised rules	60		60		138		138	
	Classification rates %	**88,17**	**1,11**	**88,00**	**1,05**	**94,58**	**1,21**	**95,75**	**1,09**
0.80	Nb of optimised rules	76		76		187		187	
	Classification rates %	**88,17**	**1,11**	**88,00**	**1,05**	**94,83**	**1,18**	**95,17**	**1,07**
0.5	Nb of optimised rules	72		72		196		196	
	Classification rates %	**88,00**	**1,01**	**88,00**	**1,01**	**95,58**	**0,99**	**95,42**	**0,99**

For the same fixed parameters, the number of generated rules is more important with the intra-classes method than with the linear correlation search upon all classes one. Indeed, in the case of the intra-classes method, the subsets of correlated components, used to build the premises of the rules, are more numerous, and so are the rules. The increase of the rules is overcomed thanks to the optimization module that eliminates the neutral rules.

A finer analysis of the performances of our system with all parameters combinations shows that the most performant co-norm is the p+q-pq one. The principle of this co-norm is to reinforce the belief degrees. In the case of the correlation search upon all classes, the co-norm p+q-pq is always better or similar to the co-norm max. In the case of an intra-classes linear correlation, the two co-norms are practically similar in all the cases.

We can refine our studies, analyzing the influence of the variation of one parameter when all the others are fixed, and comparing the two methods of linear correlation searches. The results of the correlation threshold studies are shown in Fig. 3 and Fig. 4. In these figures, we have the graphs obtained with the co-norm p+q-pq, and the two methods of linear correlation search, each figure corresponds to a subdivision cardinal tested value.

These results allow us to conclude that the method of linear correlation intra-classes is more performant than the linear correlation search upon all classes. The only cases where the results of the first method (correlation search upon all classes) are better (however almost the same) than those of the correlation intra-classes, are the cases with very good classification rates (near 96%). Changing the correlation method could not appreciably improve the results.

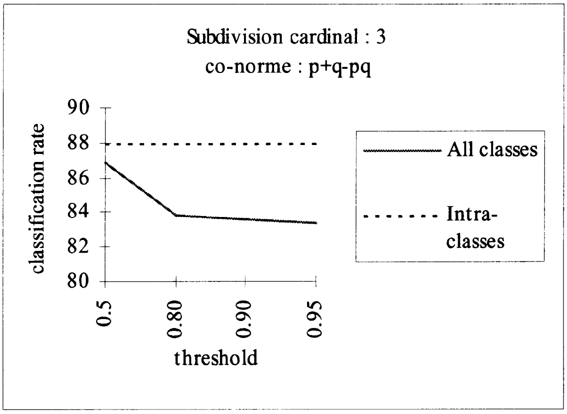


Fig. 3. Correlation threshold influence, with a subdivision cardinal of 3

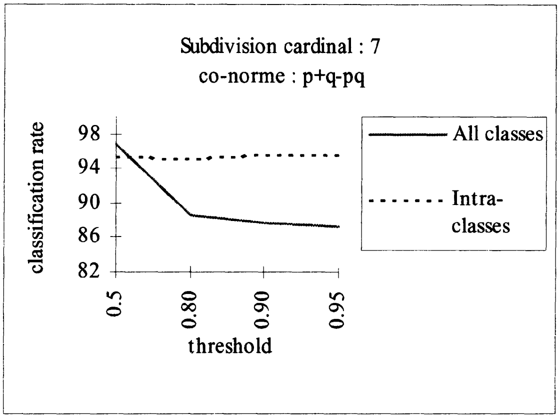


Fig. 4. Correlation threshold influence with a subdivision cardinal of 7

A linear correlation search among the vectors of each class considered separately permits to access to more discriminating informations : each class is characterized by its own matrix and correlated components. However, if the linear correlation is done among the components of all the training set vectors, without any distinction of classes, some discriminating informations are lost and the classification rates are worse. Nevertheless, we have to note, in consideration to this last method (correlation search upon all classes), that its results are satisfying, and that its complexity is weaker than the intra-classes method complexity (less matrixes to treat and less rules to generate).

7 Conclusion

The main result of our work is the improvement of the classification rates using an intra-classes linear correlation search instead of a more classical one search done upon all the classes. The classification rates improvement is accompanied by an important increase of the generated rules number. This problem was resolved by optimizing the number of the built rules. Another important point in our study is the fact that the classification rate reaches about 96% in the case of the human thigh section image.
These results are the consequences of the adequate representation of the expert's knowledge by the automatic generated rules. That means that our building system is successful in the translation of the elementary medical expert knowledge into production rules. Moreover, the classifier exploits nearly perfectly these rules, as the low error rate proves. However, these results are issued from pixels belonging to the same image. An interesting perspective is to use our system to classify pixels from different images of the same category (like other sections of thigh). Evidently, such classification systems are valid only when the image category of the training set's pixels is the same as that one of the pixels to classify. For instance, a training on satellite images can't be used to classify cartoons images.

These encouraging results lead us to test our modelization with other images and also in other fields of expertise where objects to classify can be represented by vectors.

Acknowledgments

The authors like to express their sincere appreciation to Noël Bonnet and Didier Gillard for their contribution to collect image data. They also thank Mohamed Khodja and Ilham Saker for their comments.

References

1. Vernazza, G.: Image Classification By Extended Certainty Factors. Pattern Recognition, Vol. 26, No. 11, Pergamon Press Ltd (1993) 1683-1694
2. Ginsberg, M. L.: Multivalued logics : a uniform approach to reasoning in artificial intelligence. Computational Intelligence Vol. 4, (1988) 265-316
3. Krishnapuram, R. and Keller, J.: A Possibilistic Approach to Clustering. IEEE Transactions On Fuzzy Systems, Vol. 1, No. 2 (1993)
4. Mogre, A., McLaren, R., Keller, J. and Krishnapuram R.: Uncertainty Management for Rule-Based Systems with Applications to Image Analysis. IEEE Transactions on Systems, Man, and Cybernetics, Vol. 24, No. 3 (1994)
5. Pearl, J.: Numerical Uncertainty In Expert Systems. Readings in Uncertain Reasoning. Edited by G. Shafer and J. Pearl. Morgan Kaufman publishers (1990)
6. Shortliff, E. H., Buchanan, B.G.: A Model of Inexact Reasoning in Medicine. Readings in Uncertain Reasoning. Edited by G. Shafer and J. Pearl. Morgan Kaufman publishers. Inc. California (1990)
7. Heckermans, D.: Probabilistic Interpretations for MYCIN's certainty factors. Uncertainty in Artificial Intelligence. L.N. Kanal & J.F. Lemmer Editors (1986)
8. Shafer, G.: The Art of Clausal Conjecture. Edited by the MIT Press Cambridge, Massachusetts (1996)
9. Borgi, A., Bazin, J-M., Akdag, H.: Classification supervisée d'images par génération automatique de règles. Cinquièmes Rencontres de la Société Francophone de Classification, SFC'97. Lyon (1997).
10. Borgi, A., Bazin, J-M., Akdag, H: Supervised Classification by Automatic Rules Generation. The Fourth World Congress on Expert Systems, Mexico City (1998).
11. Akdag, H., De Glas, M., Pacholczyk, D.: A Qualitative Theory of Uncertainty. Fundamenta Informaticae 17 (1992) 333-362
12. Dubois, D., Prade, H.: Théorie des possibilités, Application à la représentation des connaissances en informatique. Masson, Paris (1985)

TURBIO: A System for Extracting Information from Restricted-Domain Texts

J. Turmo, N. Català and H. Rodríguez

Dept. Llenguatges i Sistemes Informàtics*.
Universitat Politècnica de Catalunya
c/ Jordi Girona Salgado 1-3
E-08034 Barcelona - Spain

Abstract. The more extended way of acquiring information for knowledge based systems is manually. However, the high cost of this approach and the availability of alternative Knowledge Sources has lead to an increasing use of automatic acquisition approaches. In this paper we present TURBIO, a Text-Based Intelligent System (TBIS) that extracts information contained in restricted-domain documents. The system acquires part of its knowledge about the structure of the documents and the way the information is presented (i.e. syntactic-semantic rules) from a training set of them. Then, a database is created by means of applying these syntactic-semantic rules to extract the information contained in the whole documents.

1 Introduction

The more extended way of acquiring information for knowledge-based systems is manually, frequently by means of a dialog between the system and the human expert (sometimes with the intervention of a knowledge engineer). However, the high cost of this approach, together with the availability of alternative Knowledge Sources has lead to an increasing use of automatic acquisition approaches.

Special interest present the Text-Based Intelligent Systems (TBIS), in which the knowledge to be extracted is contained in documents and must be extracted from them. When these documents have been produced for computing use they use to be highly structured and extracting information from them can be carried out quite straightforwardly, but frequently the documents have been produced for human use and lack an explicit structuring. In this case they consists of an unrestricted Natural Language text and the task of extracting information involves a great deal of linguistic knowledge in order to be performed. Sometimes, and it

* This work has been partially funded by CICYT (TIC96-1243-c03-02, ITEM project) and by CIRIT (1997SCR51)

is the case of our proposal, documents present a mixed structure where chunks of NL text appear together with more structured codified pieces of information.

The techniques and methodologies for extracting information from unrestricted NL text conforms what is called Information Extraction (IE), (see [9] for an in depth survey or [3] for an introductory overview).[2]

Most IE systems are related to the MUC competitions (see [7] for a survey of MUC-6) although there are also notable examples outside. JASPER [1], FASTUS [2], LASIE [5], SRA [8], PLUM [12], NYU [6] and UMass [4] are some of the most known systems. In last years several programs have grown funded by EU. Between them ECRAN[3], AVENTINO[4] and SPARKLE[5].

TURBIO is a system, including both a methodology and a computer environment supporting it, that extracts information from semi-structured texts. TURBIO can deal with unrestricted text but also takes profit of codified pieces of information both for extracting this information and for guiding the extraction process elsewhere. The basic requirement is that a grammar could be built for allowing the extraction of codified parts and the location of chunks of NL text. TURBIO has been applied in the domain of mycology for extracting information from cards [10] describing mycological species[6]. The extracted information has been then used as knowledge base of the expert system KINOKO[11] that classifies unknown species from their features.

2 Functionalities

A main issue in IE systems is the definition of the extraction rules. Generally, extraction rules are represented by pairs <keyword,template-set>, where *keyword* refers to a domain concept and each template of the *template-set* represents the set of modifier features for that concept. Extraction rules are usually defined manually. TURBIO proposes, however, the acquisition of extraction rules using a learning process based on an analysis of a training corpus. The resulting rule set (containing the knowl-

[2] IE is an emerging technology and must not be confused with the more mature area of Information Retrieval (IR), that giving a query, tries to select a relevant subset of documents from a larger set.

[3] http://www2.echo.lu/langeng/en/lel/ecran/ecran.html

[4] http://www2.echo.lu/langeng/en/lel/aventinus/aventinus.html

[5] http://www2.echo.lu/langeng/en/lel/sparkle/sparkle.html

[6] Although our corpus consists of bilingual (Spanish/Catalan) non-parallel descriptions, only texts in Spanish have been used here.

edge about corpus structure and the way of describing information) will be then used to perform the information extraction process.

The first functionality of TURBIO is then to extract patterns of relevant information contained in texts belonging to specific domains using a domain structured representation. The basic units to be extracted are triples <entity attribute value>.

The second functionality consists of applying these rule sets for extracting information from documents. Briefly, once the text has been preprocessed and shallow-parsed, the system looks for a keyword and its modifiers in the parse forest and activates the associated template covering more modifiers.

Following, we explain the TURBIO methodology to get both functionalities.

3 Architecture

Figure 1 presents an overview of TURBIO architecture. TURBIO builds a structured representation of the documents (DB)[7] using a grammar for describing the document structure. A representation of the domain must be provided too.

The result of TURBIO performance is a domain knowledge base (DKB) represented in a typed feature structure formalism containing instances of entities of the domain owning the information extracted from texts.

The first functionality of TURBIO -extraction rule set acquisition- is performed by three modules of the system: S-BUILDER, P-BUILDER and KERNEL. S-BUILDER gets chunks of shallow-parsed trees, generalizes them in *syntactic-semantic pattern schemata* using GENERALIZER module and produces the set of relevant pattern schemata in the corpus. P-BUILDER is used to build relevant *syntactic-semantic patterns* from schemata trying to fix variables in a schema with common information of the chunks it represents. Finally, KERNEL builds the extraction rule set.
The extraction functionality of TURBIO is performed by KERNEL. It produces feature structures by applying extraction rules over the shallow-parsed chunks.

Next sections gives details of all these modules.

[7] Part of DB is used as training corpus during the learning process.

4 S-BUILDER module

The aim of this module is to extract all relevant syntactic-semantic pattern schemata occurring in the corpus. An schema means a representation of the set of chunks and subchunks having the same syntactic-semantic parse tree. S-BUILDER runs in three steps: 1) a pre-process in which morphological and shallow syntactic parsing is performed for getting shallow-parsed chunks of phrases, 2) a process to find syntactic-semantic pattern schemata from those chunks (GENERALIZER module) and 3) a process to select from them relevant pattern schemata. For instance, in the pre-process, the phrase:

> *"Crema amarillento y de carne oscura."* (Yellowish cream with dark flesh)

is morphologically parsed as:

> n(*"crema"*) a(*"amarillento"*) cOc(*"y"*) rOa(*"de"*) n(*"carne"*) a(*"oscura"*) zpunt(*"."*)

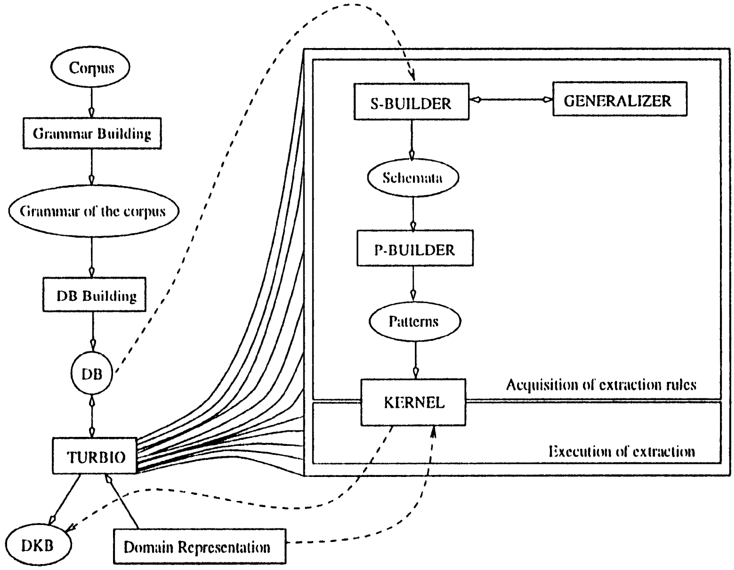


Fig. 1. Environment and architecture of TURBIO

and shallow parsed as:

1. (gnom ((n *"crema"*)(a *"amarillento"*)))
2. (cOc *"y"*)
3. (sp ((rOa *"de"*)(gnom ((n *"carne"*)(a *"oscura"*)))))
4. (zpunt ".")

where each line is a chunk and each subtree of a chunk is a subchunk.

Analyzing these chunks we find the pattern schemata:

a. (gnom ((n [R_1:S_1]-W_1) (a [R_2:S_2]-W_2)))
b. (sp ((r0a W_1) (gnom ((n [R_1:S_1]-W_2) (a [R_2:S_2]-W_3)))))

(a) represents both the chunk (1) and the second subchunk of chunk (3), where W_i contains the possible words in the schema (e.g. W_1={ *"crema" "carne"*}), S_i the list of semantic values for W_i and R_i relations between semantic values and words[8].

4.1 Defining δ-sets

In order to find relevant patterns some relationships between schemata must be introduced. Schemata have been coded assigning to each syntactic label a prime number and multiplying the codes of each syntactic label in the schema (δ-code). For example, the schema:

r =(gnom ((n [R_1:S_1]-W_1) (a [R_2:S_2]-W_2)))

has $\delta(r)$ = 30 when we codify *gnom* as 2, *n* as 3 and *a* as 5. This coding allows to define the equivalence relation $=_\delta$ between schemata as follows:

Let S *the schemata set,*
$r, s \in S$, δ(r)= δ(s)$\rightarrow$r$=_\delta$s

Now it is possible to define the quotient set $S|_{=_\delta}$. In this way, a δ-set will be an element of $S|_{=_\delta}$. For example, both schemata:

r =(gnom ((n [R_1:S_1]-W_1) (a [R_2:S_2]-W_2)))
s =(gnom ((a [R_1:S_1]-W_1) (n [R_2:S_2]-W_2)))

belong to the same δ-set.

4.2 Finding relevant pattern schemata

This module finds the relevant set of pattern schemata. As we can see, schema (1) of the example of this section is included in schema (2). It is possible to define a relation between schemata, called *covering* relation:

$r, t \in S$, t *covers* $r \leftrightarrow r$ *covered by* $t \leftrightarrow t \sqsubseteq r \leftrightarrow t$ *includes* r

We can classify schemata in two classes: those derived from chunks and those derived from subchunks (Fig. 2). The intersection of both sets is not empty. In order to get the set of relevant schemata we must study those pairs of high frequency schemata covering-related. Then, all high frequency schemata not covered by any other will be relevant.

[8] Only names, adjectives and verbs own R_i and S_i.

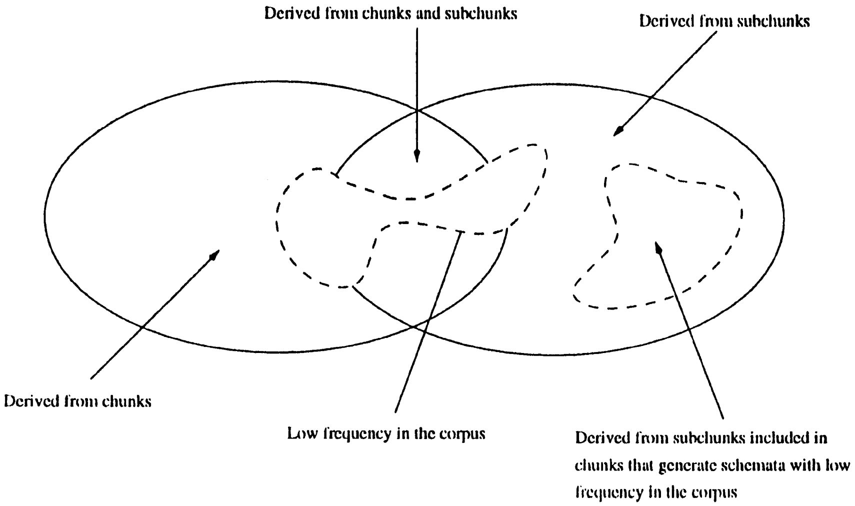

Derived from chunks and subchunks
Derived from subchunks
Derived from chunks
Low frequency in the corpus
Derived from subchunks included in chunks that generate schemata with low frequency in the corpus

Fig. 2. Classification of schemata according to their derivation

We can represent covering relation between schemata using the δ-sets defined in section 4.3. A link relation exists between pairs of δ-sets when there is, at least, one schema in one of them covered by another schema of the other δ-set (linked δ-sets). This link is verified by: two δ-sets δ_a, δ_b are linked-related iff $\delta_a \equiv_{\delta_b}$.

In figure 3 δ-sets 30 and 210 are linked because schema t covers schema r being 210 module 30 equal to zero. So, the method to get relevant schemata consists of:

- Drop out schemata without nominal, verbal or adjectival labels.
- Drop out schemata with low frequency in the corpus ($f_s < 10$).
- Lessening of δ-sets: residual frequency of a schema is the difference between its frequency in the corpus and the sum of frequencies in the corpus of all other schemata covering the first one. The lessening of δ-sets means to drop out all schemata having null residual frequency. We are selecting only schemata uncovered by more specific ones.

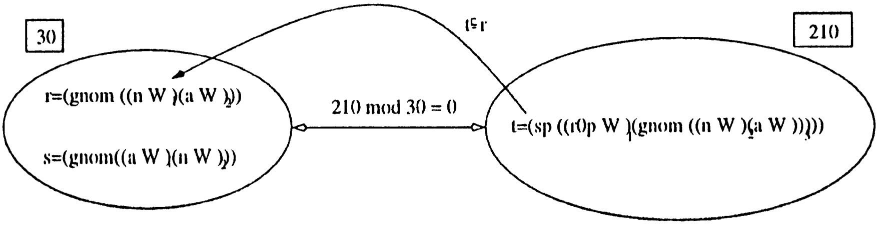


Fig. 3. Linked δ-sets

4.3 GENERALIZER

Chunks found can be seen as specific syntactic patterns. Each constituent of the pattern matches specific words (e.g., *"pie"*, *"blanco"*), and specific syntactic properties of these words (e.g., n(ame), a(djective)).

Observing the set of specific patterns we note that two or more patterns, having the same syntactic restrictions, have also semantic properties in common but different specific words to apply to. This suggests that these patterns can be generalized using semantics.

Such assumption implies that we have semantic knowledge about our language or, at least, about the sub-language the application domain deals with. Our approach makes use of an Spanish WordNet (part of EuroWordNet)[9] as a basis of semantic knowledge because its representation as a semantic net allows us to reason about different types of semantic relationships between concepts. In addition, for simplicity, we consider that a conceptual representation corresponding to the domain vocabulary is available.

Figure 4 shows, partially, the conceptual representation in WordNet and the label meanings (relationships) of an abbreviated mushrooms vocabulary.

The generalization of two specific patterns starts looking for a more general concept covering both concepts. We need that all constituents could be generalized at some level of the hierarchy. If it is not possible, the specific patterns are maintained.

Since each specific concept may have several generalizations using a variety of relationships, a method for reducing the problem is needed. The search of the ancestor concept moving up in the hierarchy is guided by different rules depending on the relationships present in the domain semantic representation.

The easiest way of obtaining ancestors is provided by the *hyperonymy* relationship (and its homonym relation across different *part of speech*, *xpos-hyperonymy*) but there are also other relationships, such as the variety of types of *meronymy*, that allow generalization if they are controlled. Some relations shown in Fig. 4, play the role of restrictions with regard to the application of generalization rules. For instance, *hyperonymy* relationship is restricted by the existing *has-property* relationship between two concepts, while generalizing them.

[9] http://www.let.uva.nl/~ewn

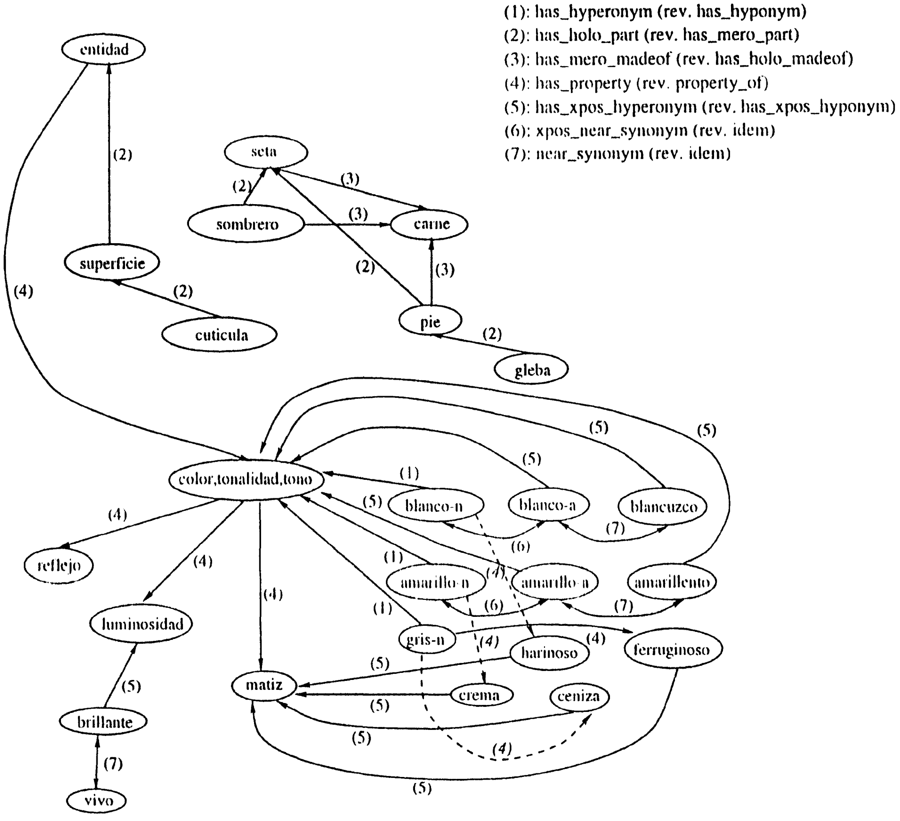

(1): has_hyperonym (rev. has_hyponym)
(2): has_holo_part (rev. has_mero_part)
(3): has_mero_madeof (rev. has_holo_madeof)
(4): has_property (rev. property_of)
(5): has_xpos_hyperonym (rev. has_xpos_hyponym)
(6): xpos_near_synonym (rev. idem)
(7): near_synonym (rev. idem)
entidad
seta
sombrero
carne
superficie
cuticula
pie
gleba
color,tonalidad,tono
blanco-n
blanco-a
blancuzco
reflejo
amarillo-n
amarillo-a
amarillento
luminosidad
gris-n
harinoso
ferruginoso
matiz
crema
ceniza
brillante
vivo

Fig. 4. Spanish WordNet (abbreviated mushrooms sub-language).

The next example shows partially the generalization process applied to a set of specific patterns provided by the previous module (S-BUILDER).

(gnom ((n *"crema"*) (a *"amarillento"*)))
(gnom ((n *"gris"*) (a *"ferruginoso"*)))
(gnom ((n *"pie"*) (a *"blanco"*)))
(gnom ((n *"blanco"*) (a *"harinoso"*)))
(gnom ((n *"ceniza"*) (a *"blancuzco"*)))
(gnom ((n *"sombrero"*) (a *"rosado"*)))

Suppose that the chunk (specific pattern) list above is a complete list of all chunks built from the test text. Starting with the first two chunks there is no generalization possible because semantics of *"crema"* (that is a MATIZ) and semantics of *"gris"* (that is a COLOR) doesn't have a common hypernym concept in any level of the hierarchy.

Between the first three chunks there is no generalization possible for similar reasons. But when we consider the fourth one, a generalization is possible between chunks two and four. Semantics of *"gris"* (that is a COLOR) and semantics of *"blanco"* (that is a COLOR) have a common concept in the hierarchy, and semantics of *"ferruginoso"* (that is a MATIZ) and semantics of *"harinoso"* (that is a MATIZ) have also a common concept in the hierarchy. We represent the resulting generalized pattern as follows:

(gnom ((n [HYPONYM:COLOR]-W_1) (a [HYPONYM:MATIZ]-W_2)))

Following the example it is also possible a generalization between chunks one and five, and between three and six. The former generalization results in:

(gnom ((n [HYPONYM:MATIZ]-W_1) (a [HYPONYM:COLOR]-W_2)))

and the last generalization results in:

(gnom ((n [ISPARTOF:SETA]-W_1) (a [HYPONYM:COLOR]-W_2)))

Finally, each generalized pattern will be converted into its representation as a syntactic-semantic pattern schema, i.e. changing concepts and relationships for variables and maintaining the original information using linked sets (LSs) as explained above.

Different rules are applied on different specific patterns in order to generalize them and the resulting generalizations could be too general for the purposes of the type of information to be extracted. We need a domain-expert detecting the over-generalization and setting a limitation on the application of rules.
Once finished the S-BUILDER process, relevant schemata are used by P-BUILDER module.

5 P-BUILDER module

The aim of P-BUILDER module is to get syntactic-semantic patterns from the relevant schemata found by S-BUILDER. Patterns are defined as specializations of schemata.

As we saw in section 2, the basic units to extract are <entity attribute value> triples. Patterns can be classified into generic and specific ones, depending on the information they own, and into simple and compound according to the way they have been generated.

Extraction of simple patterns is done by analyzing LSs associated with schemata:

- All schemata with invalid LSs for the domain are dropped out.
- Some patterns are generated combining words, relations and semantics with high frequency in the corpus using previously selected schemata. In the case of the schema:

  (gnom ((n [R_1:S_1]-W_1)(a [R_2:S_2]-W_2)))

  it generates patterns like:

  p_1=(gnom ((n [ISA:CARNE]:W_1)(a [HYPONYM:COLOR]-W_2)))

  where p_1 is a pattern for *"carne amarillenta"* (*"yellowish flesh* ").
- The rest of possible specializations are rejected because, in general, they represent patterns out of the restricted domain or false patterns because of the ambiguity of words. GENERALIZER may produce more than one generalization. But only the most frequent patterns for a schema will be selected. For example *"castaño"* has two senses: the chestnut color and the chestnut tree. The presence of the second sense of *"castaño"* in our domain and in the Spanish WordNet as a hyponym of the concept *habitat* could produce the pattern:

  p_3=(gnom ((n [ISA:CARNE]:W_1)(a [ISA:CHESNUTTREE]-W_1)))

  This false pattern must be rejected.
- Finally, simple patterns used only by finding compound ones are rejected.

Once simple patterns have been extracted, P-BUILDER generates compound patterns taking into account combinations of simple ones. For example, it is frequent to find next simple patterns together in the corpus:

p_4=(sp ((r0a *"de"*)(gnom((n [HYPONYM:COLOR]-W_1))))
p_5=(sp ((r0a *"a"*)(gnom((n [HYPONYM:COLOR]-W_1))))

So, they produce the compound pattern $p_4 + p_5$ to represent linguistic expressions of the *color* concept as interval value (e.g. *"de verde a amarillo"*).

6 KERNEL module

With KERNEL the system builds extraction rules using two sub-modules: PC (Priority Classifier) and ERG (Extraction Rules Generator) (Fig. 5).

PC module classifies patterns in a hierarchy of priorities according to: 1) their specificity and 2) the length of the pattern. Then ERG module builds extraction rules. The condition of a rule is a pattern and the action is the extraction method needed for that pattern. A set of methods, one for each class of pattern, has been built. Currently KERNEL includes five methods.

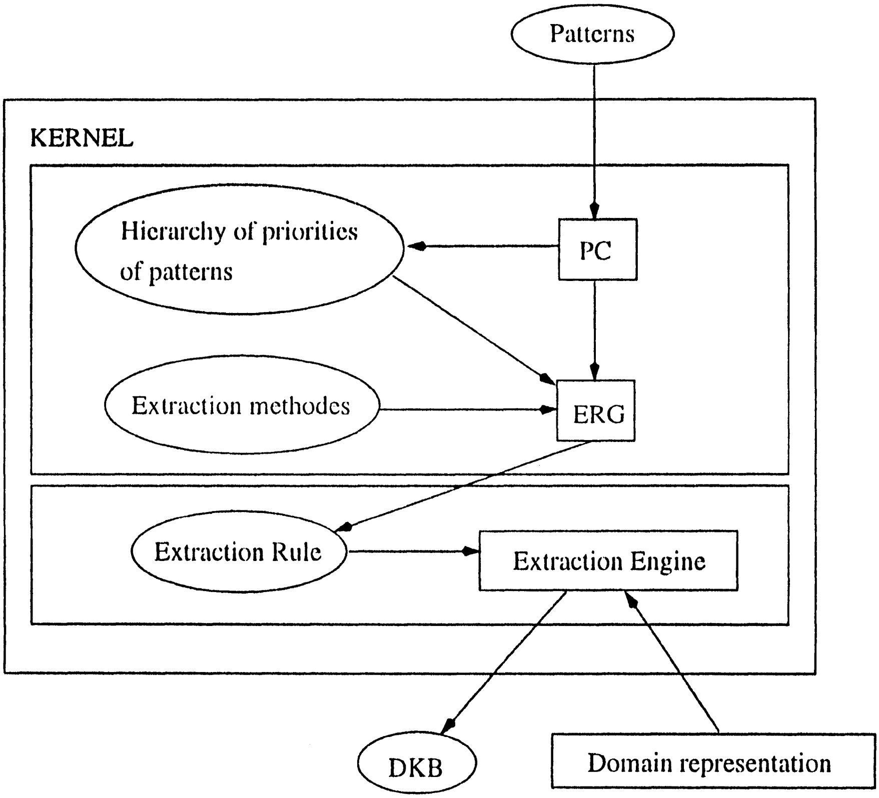


Fig. 5. Architecture of KERNEL module

The third component of KERNEL, the Extraction Engine, can operate in isolated way in order to carry out the extraction task.

7 Results

TURBIO has been tested using 150 mycological cards as training corpus. This represents a total of 21609 words (2991 different lemmas). We have reduced the test to the study of the *color* attribute. Results can be generalized because the color comprises all possible classes of linguistic expressions supported by other attributes. Being Spanish WordNet under construction we have applied the GENERALIZER module only to selected examples. Results reported here have been obtained using only syntactic information.

The S-BUILDER module generated 159 schemata when obtaining δ-sets. 35 of them were refused because they did not contain any valid label, 73 were dropped out due to low frequency in the corpus and 2 in lessening of δ-sets. So, we obtain 49 relevant schemata.

In P-BUILDER module 4 schemata were rejected due to invalid LSs for the domain. From the rest of schemata only 28 were referred to the *color* attribute. With the P-BUILDER methodology we found 53 simple patterns and 68 compound ones, 7 simple patterns of them were dropped out because they only were used to generate compound ones.

In total we found 114 relevant patterns. All of them were used to get extraction rules by the KERNEL module.

The results of the extraction were 48.30% for *recall* and 87.14% for *precision*[10]. Uncoverage is analyzed in the next table:

Cause	Uncovered cases	Partial Cause
No extraction rule	11.49% 16.26%	Not extracted Partially extracted
Erroneous POS tagging	12.46%	
No coreferent found	6.22% 5.27%	Elision without coreferent in the context Value with reference to other entity

Results are acceptable but not easy to evaluate because there is no valid benchmark for Spanish (as MUC for English).

[10] *Recall* is the percentage of possible answers which were correct.*Precision* is the percentage of actual answers given which were correct.

8 Conclusions and future work

In this paper we have presented a system for extracting information from restricted domain texts using extraction rules learnt from a sample subset of the corpus.

In most existing IE systems the extraction rules are manually provided. Some others use a semi-automatic approach allowing a human expert the selection of the proper level of syntactic-semantic generalization from parse-chunks. Our approach minimizes human intervention using shallow parsing over the whole document and limiting the manual task to validate the automatic semantic generalization.

TURBIO has been tested in mycological domain getting acceptable levels of precision and recall.

Future work includes two main lines:

- The semantic component of TURBIO must be extended for covering a substantial amount of the vocabulary.
- The high error-rate of our tagger is mainly due to the specificity of the sub-language and the high number of unknown words. These problems will be approached tuning the tagged with a specific domain corpus and including a module for dealing with unknown words.
- A bidirectional island-driven shallow parser will be used to deal with the reference problem when necessary.
- We plan to apply TURBIO to English texts in order to compare its performance with widely used benchmarks.

References

1. P.M. Andersen, P.J. Hayes, A.K. Huettner, I.B. Nirenburg, L.M. Schmandt, S.P. Weinstein. *Automatic extraction of facts from press releases to generate news stories.* In Proceedings of Third Conference on Applied Natural Language Proceeding, pages 170-177. Association for Computational Linguistics, 1992.
2. D. Appelt, J. Hobbs, J. Bear, D. Israel, M. Kameyama, A. Kehler, D. Martin, K. Meyers, M. Tyson. *SRI International FASTUS system: MUC-6 test results and analysis.* In Proceedings of 16th MUC, Columbia, MD. 1993.
3. Ed. J. Cowie, W. Lehnert. *Information Extraction. Special Issue.* Communication of ACM. Vol.39(1). 1996.

4. D. Fisher, S. Soderland, J. McCarthy, F. Feng, W. Lehnert. *Description of UMass system as used for MUC-6.* In Proceedings of 16th MUC, Columbia, MD. 1996.
5. R. Gaizauskas, T. Wakao, K. Humphreys, H Cunningham, Y. Wilks. *Description of the LaSIE System as used for MUC-6.* In Proceedings of 16th MUC, Columbia, Maryland, USA. 1995.
6. R. Grishman. *The NYU system for MUC-6 or where's the syntax?.* In proceedings of 16th MUC, Columbia, Maryland, USA. 1995.
7. R. Grishman, B Sundheim. *MUC-6. A brief history.* In Proceedings of 16th Int'l Conf. on Computational Linguistics. 1996.
8. G. Krupka. *Description of the SRA system as used for MUC-6.* In Proceedings of 16th MUC, Columbia, MD. 1995.
9. Ed. M. Pazienza. *Information Extraction. A Multidisciplinary Approach to an Emerging Information Technology.* Springer. 1997.
10. Societat Catalana de Micologia. *Bolets de Catalunya.* ISSN 0212-3460, Ed.: Societat Catalana de Micologia. Facultad de Farmacia de Barcelona. 1994.
11. J. Turmo. *KINOKO: Sistema Experto para la Clasificacion de Micetos basado en Hechos.* Master thesis in Computer Science for the Universitat Politècnica de Catalunya. 1996.
12. R. Weischedel. *Description of the PLUM system as used for MUC-6.* In Proceedings of 16th MUC, Columbia, MD. 1996.

An Approach to Ellipsis Detection and Classification for the Arabic Language

Kais Haddar[1] and Abdelmajid Ben Hamadou[1]

[1] FSEGS Sfax, Laboratory LARIS, University of Sfax, B.P. 1088,
3018 Sfax, Tunisia
Abdelmajid.Benhamadou@fsegs.rnu.tn

Abstract. The phenomenon of ellipsis is particularly difficult to be treated automatically because it isn't usually easy to characterise the elements that we can delete. In this present article, we propose a formal characterisation of the phenomenon of ellipsis and a localization algorithm of elliptical parts of a sentence based on a clause parser. Then, we present a certain number of classification criteria of elliptical sentences to prepare the phase of resolution itself later.

1 Introduction

Ellipsis has always been a center of academic interest, be it in linguistic theory or in computational linguistics. A simple example of Arabic ellipsis is given in sentence (1):

ذهب الولد الصغير الى المدرسة و [ذهب] أخوه الى الجامعة. (1)

The young boy went to school and his brother [*went*] *to university.*

In this example, the second clause contains the deletion of a verbal group, the meaning of which is to be determined from the first clause. According to Dalrymple terminology [4], we call the clause that the verb is copied from the well-formed clause, and the clause which contains the elided verb the elliptical clause.

Syntactic accounts of ellipsis posit the copying of syntactic structure from the well-formed clause representation to the elliptical clause one [5], [7] and [10].

In this paper, we take an interest in the automatic treatment of the phenomenon of ellipsis in discourse written in Arabic language. To identify the well-formed clause in an elliptical sentence and to localize the elliptical ones, we propose to construct an Arabic grammar that recognises only the structure of well-formed clauses. Also, the grammar can be tightly restricted to define the sentence structure in terms of a small number of well-formed clause structures.

During the Arabic clause grammar construction, we try to define:

- The larger constituents of a well-formed clause. For example, what noun phrases, verb phrases, prepositional phrases occur in the clause.
- The way how the constituents should be combined or attached to form the overall syntactic structure of the well-formed clause.

The originality of our approach resides in using a clause grammar to detect ellipses of complex sentence structures and in establishing a classification of elliptical sentences which facilitates the construction of efficient resolution algorithms [6].

The paper is structured as follows: in section 2, we present some related works and compare them to our approach. Section 3 is devoted to the presentation of the clause grammar used for the well-formed clause identification. In section 4, we give an ellipsis detection method based on a formal characterisation of elliptical sentences. In section 5, we propose an elliptical sentence classification based on the syntactic structure of the elliptical sentence. The last section presents our conclusions and futures works.

2 Related Work

Several methods have been used to treat ellipses in their different forms and in different contexts. We cite in particular:

- The extension of the grammar by adding explicit rules or metarules that prevent ellipses [5], [8] and [14].
- The relaxing of constraints: Consistency and ordering constraints can be relaxed on a second pass, allowing the detection and resolution of ellipses [15].
- The fitted parsing: It happens when the rules of a conventional syntactic grammar are unable to produce a parse for an elliptical sentence. This technique can be used to produce a reasonable approximate parse that can serve as input to remaining stages of natural language processing [9].

Traditional linguistic theory distinguishes ellipsis forms (i.e., Right-node Raising, Gapping, VP-deletion, Coordinate Reduction) according to syntactic structure of elliptical clauses [5]. Syntactic accounts of these ellipsis forms posit the copying of syntactic structure from the well-formed clause representation to the elliptical clause one. Other approaches involve the semantic representation of an appropriate property or relation for the elided constituent fragment [4]. All these approaches are generally based on a complex sentence grammar that can not easily locate the elliptical parts of the sentence [3], [5] and [8]. In these approaches, elliptical clause occurrence does not play any role. In our approach, we distinguish ellipsis classes according to syntactic structure of elliptical sentences by taking into account the following considerations:

- occurrence of elliptical clauses that a sentence contains,
- alternation of well-formed clause/elliptical clause,
- ellipsis form.

Unlike the above mentioned approaches, our approach is based on another type of grammar namely an ATN clause grammar used to identify the well-formed clause and to locate the elliptical ones. It can be classified as a syntactic account of ellipses.

3 Clause Parser

Syntactic analysis can be performed using complex sentence grammar. But, the specification of such a grammar is complex and requires much effort. Besides, it will be difficult to localise the elliptical parts of the sentence. Our proposition is to use a clause grammar. So, we minimize the effort required in writing the sentence grammar, we can easily identify the well-formed clause of an elliptical sentence and then locate the elliptical ones, and we can detect the elliptical fragments of an elliptical clause. Note that an Arabic sentence may begin either with a verb phrase (VP) or with a noun phrase (NP). We restrict our study to the sentences that begin with VP. As in [11], we consider an exception phrase fragment not as an instance of ellipsis, but as a displaced NP modifier. In the following, we give a subset of Arabic grammar rules:

S → VP + NP
S → VP + (NP|PP) + exception prep + NP
S → interrogative pro + VP + NP
S → S + (NP|PP)
NP → NP + conj + NP | NP_1 | NP_2 | NP_3 | NP_4 | noun | proper noun | pro
PP → prep + (NP_1| NP_2| NP_3| NP_4| noun)
NP_1 → art + noun | demonstrative pro + noun
NP_2 → NP_1 + AP
NP_3 → noun + (NP_1|NP_2)
NP_4→ noun + adj | NP_4+ adj
VP → negative adv + verb | verb
AP → art + adj | art + adj + AP.

The clause grammar that we use for our approach is an augmented context-free clause grammar. To implement it, we construct first a Recursive Transition Network (RTN) [16]. Then, we transform it into an ATN in order to add subcategorization principle and agreement constraints which are useful for the resolution process and to avoid ambiguity.

The major components of our clause parser are:

- a set of lexical categories for the words of the Arabic language and a lexicon in which each word is assigned the categories that apply to it [1];
- an Arabic clause grammar, using the same categories as the lexicon and whatever constructs are required by the type of grammar used to specify the well-formed clause structures;
- a parsing program, whose inputs are the clause to be processed, the lexicon, and the Arabic clause grammar, and whose output for each clause is a structural description of the clause.

4 Ellipsis Detection Method

We begin by recalling that it is possible, from the linguistic viewpoint, to adopt for the Arabic language the same typology of ellipses (i.e., Gapping, Right-node Raising, Coordination Reduction,...) as the one proposed for English and French [5], [7], [10] and [13]. In the following, we present the main characteristics of ellipses in the Arabic language.

4 . 1 Typology of Ellipsis in Arabic Language

As in other natural languages, ellipsis is the omission of one or more words in a sentence that remains however comprehensible. These words will be necessary only

for the regular construction of the sentence. Generally, the meaning of the sentence is completed by reference to a part of the previous discourse.
Example.

استعملت الحاسوب كذلك [استعمل] بقية الطلبة [الحاسوب]. (2)

I used the computer and the other students did [*use the computer*] *so.*

Words which are omitted are bracketed.

We give hereafter a typology of ellipses of the Arabic language inspired from that proposed in [12] and [14].

Set Forms of Ellipses. Certain expressions, and notably letter endings, forms of wishes, certain orders or certain exclamations are elliptical. To understand them, it isn't necessary to construct the complete form. That's why, they are called false ellipses.
Examples.

Happy new year [أتمنى لك] عيدا سعيدا (3)

Fire, fire [أحذر] النار, النار (4)

These set forms of ellipses do not interest us because they can be resolved at the lexical level. But we rather focus our interest on the ellipses that oblige the reader to search in the context the lacking elements without which the message will be incomprehensible. The sentences (1) and (2) are some examples of this.

Nominal Ellipsis. The nominal ellipsis is the omission of the essential part of a noun phrase (i.e., head) that has to be taken from outside of the part of the previous discourse. Then in a sentence that contains a nominal ellipsis, one of the noun phrases is incomplete.
Example.

الأخ الأكبرله من العمر عشرسنين و [الأخ] الأصغر ستة سنوات. (5)

The oldest brother is twenty and the youngest [*brother*] *is six years old.*

In example (5), the nominal phrase containing ellipsis is represented by an article and an adjective (the youngest). This treated example contains a nominal ellipsis.

Whole Phrase Ellipsis. The whole phrase ellipsis is distinguished from the nominal one by the fact that the omitted constituent may be a whole phrase (noun phrase or verb phrase). Therefore, in this category, we can distinguish the following ellipsis types:

- Gapping:

ذهب الولد الصغير الى المدرسة و [ذهب] أخوه الى الجامعة. (1)

The young boy went to school and his brother [*went*] *to university.*

- Right-node Raising:

بعت انا دراجة و اعطيت [انا] أختي صنارة. (7)

I sells a bike and [*I*] *gives a fishing rod to my sister.*

- Coordination Reduction:

طلبت من أخي دراجته و [طلبت] من أختي صنارتها. (8)

I ask my brother for his bike and my sister for her fishing rod.

4 . 2 Formalism

According to what we have exposed above and the fact that we use an ATN clause grammar, we consider that an elliptical sentence in two distinct parts: an elliptical part preceded by a well-formed part that can be used as reference to resolve the ellipsis. The elliptical part might be composed of a succession of elliptical clauses generally separated with connection words (that we call *connectors*) which are coordinators, correlative conjunctions, subordinators, ...

To formally characterise this phenomena, we must first introduce some notations and definitions.

Elliptical Clause. Let V be the finite Arabic alphabet. Let V^* be the set of finite words over V. $V^+ = V^* \backslash \{\varepsilon\}$, where ε is the empty word.

Definition 1. We call *clause* any finite sequence P of words of V^+, $P=w_1w_2...w_m$, $w_i \in V^+$ and $1 \leq i \leq m$. We note $\mathcal{P}$ the set of all clauses that can be constructed from V^+.

Let G be the Arabic clause grammar over V^+ describing thus a subset $L(G) \subseteq \mathcal{P}$. To verify whether a given clause P is generated by G, we introduce the following function that represents the acceptance function of the clause parser:

$$analyse : \mathcal{P} \rightarrow \mathcal{B} = \{true, false\}$$
$$P \mapsto \begin{cases} true \text{ if } P \in L(G) \\ false \text{ otherwise.} \end{cases}$$

Definition 2. A clause P is called *well-formed* if P is in L(G).

$$\boxed{P \textit{ well-formed} \Leftrightarrow \textit{analyse}(P)}$$

Note $\mathcal{S}$ the set of finite sequences of clauses of $\mathcal{P}$.

Definition 3. Let $S=P_0P_1...P_n \in \mathcal{S}$. We define the context of a clause P_i, $1 \leq i \leq n$ by the subsequence $S_i=P_0P_1...P_{i-1}$ formed from all clauses preceding P_i. We note by W^{Si} the set of all the words composing S_i.

Remark. The context of P_0 is the empty sequence.

Definition 4. Let $S=P_0P_1...P_n$ be a sequence of $\mathcal{S}$ and $P_i = w_{i1}w_{i2}...w_{im}$ a clause of $\mathcal{P}$, $0 \leq i \leq n$. Let the function *is_recoverable* be defined by:

$$is\_recoverable(P_i, S) = \begin{cases} true \;\; if \; \neg \; analyse(P_i) \wedge W^{Si} \neq \varnothing \wedge \exists \, w'_{i1}, \ldots, w'_{i(m+1)} \in W^{Si} \cup \{\varepsilon\} \\ \qquad \wedge \; analyse\left(w'_{i1}w_{i1}w'_{i2} \ldots w'_{im}w_{im}w'_{i(m+1)}\right) \\ false \;\; otherwise. \end{cases}$$

A clause P_i, $1 \leq i \leq n$, is called *elliptical* if *is_recoverable* (P_i, S).

Therefore, a clause P_i from a sequence S is elliptical when it isn't well-formed and when it can be completed with words of its context in order to be a well-formed one.

Remark. Examples as *he is going out and it raining* are treated like elliptical clauses.

Elliptical Sentence. Let C be a subset of V^* that we call set of connectors containing the empty word ε too.

Definition 5. A sentence Ph is a sequence of clauses separated with connectors of C. Ph has the following form:

$$\boxed{Ph = P_0c_1P_1 \ldots c_nP_n \text{ where } c_i \in C,\ P_i \in P.}$$

Definition 6. A sentence Ph is called elliptical if

1) the first clause of Ph is well-formed
2) at least one of the remaining clauses is elliptical.

Example 1. Let the sentence Ph be:

يجب ربط السلك الازرق بالنهائي أ و [يجب ربط] السلك الاصفر بالنهائي ب او [يجب ربط]
السلك الاسود بالنهائي ج.

The blue cable must be connected to terminal A and either the yellow cable [must be connected] to terminal B or the black cable [must be connected] to terminal C.

P_0: The blue cable must be connected to terminal A.
P_1: the yellow cable to terminal B.
P_2: the black cable to terminal C.
Ph is elliptical because P_0 is well-formed and P_1 and P_2 are elliptical.

4.3 Outline of the Detection Method

Basing on this formalism, we propose an algorithm for elliptical clause location in a sentence consisting of three main stages performed by the clause parser:

- **Search of connectors.** We search for all existent connectors in the analysed sentence relying on a connector lexicon. The obtained result is made up of a list containing clauses and of an other one containing connectors.
- **Identification of the well-formed clause used as reference.** We make a filtering operation of the already obtained lists of clauses and of connectors based on the identification of the longest well-formed clause of them. Once the well-formed clause is identified, the two lists are updated by removing the well-formed clause components from them.
- **Labelling of the remaining clauses.** We attribute to each clause of the remaining clause list an etiquette meaning that the clause is either elliptical or not. Labelling allow us to localise the elliptical clauses of the list.

Therefore, in order to have an elliptical sentence, it must begin with a well-formed clause and it must contain at least one clause labelled by the function *etiquette*.

At the end of those three stages, we dispose of enough information to classify elliptical sentences and later to establish resolution algorithms [6].

5 Elliptical Sentence Class Determination

We can distinguish different classes of elliptical sentences by taking into account the following considerations:

- occurrence of elliptical clauses that a sentence contains,
- alternation of well-formed clause / elliptical clause,
- ellipsis type.

5.1 Formalism

We distinguish the following class definitions:

Definition 7. An elliptical sentence $Ph_e = P_0c_1P_1 \ldots c_nP_n$, $n \geq 1$, is called *homogeneous* if all clauses P_i, $1 \leq i \leq n$, are elliptical and have the same type of ellipsis.

Example 2. Take the elliptical sentence of example 1. Since P_1 and P_2 have the same type of ellipsis then Ph is homogeneous.

Definition 8. An elliptical sentence $Ph_e = P_0c_1P_1 \ldots c_n P_n$, $n > 1$, is called *heterogeneous* if every two successive elliptical clauses P_i et P_{i+1}, $1 \leq i < n$, have not the same type of ellipsis.

Important remark. An elliptical sentence that does not contain two successive elliptical clauses is systematically heterogeneous whatever the type of ellipsis of clauses that the sentence holds.

Example 3. Let the sentence Ph be:

يجب ربط السلك الازرق بالنهائي أ و [يجب ربط] السلك الاصفر بالنهائي ب او يجب عليك

ترك النهائيات مفتوحة.

The blue cable must be connected to terminal A and the yellow cable [must be connected] to terminal B or you must keep them free.

P_0: The blue cable must be connected to terminal A.
P_1: the yellow cable to terminal B.
P_2: you must keep them free.
Ph is heterogeneous because P_0 and P_2 are well-formed and P_1 is elliptical.

Definition 9. Let $Ph_e = P_0c_1P_1 \ldots c_nP_n$, $n > 1$ be an elliptical sentence. Ph_e is called *mixed* if it is neither homogeneous nor heterogeneous.

Illustrative example. We note here by P_i a not elliptical clause and by P_i^{tj} an elliptical clause of type t_j. Let:

$Ph_1 = P_0c_1P_1^{t1}c_2P_2^{t1}$,
$Ph_2 = P_0c_1P_1^{t1}c_2P_2^{t2}$ et $Ph_3 = P_0c_1P_1^{t1}c_2P_2$,
$Ph_4 = P_0c_1P_1^{t1}c_2P_2^{t1}P_3c_4P_4^{t2}$.

According to the definitions 7, 8 and 9, Ph_1 is homogeneous, Ph_2 and Ph_3 are heterogeneous and Ph_4 is mixed.

All these definitions, which we have presented, will allow us to establish criteria to apply for the membership class determination of the elliptical sentences with the intention to perform the resolution process.

5 . 2 Heuristic

According to definitions 7, 8 and 9, in order to determine the class of an elliptical sentence, we have to identify the types of ellipses of the successive elliptical clauses. Let's recall that an elliptical sentence is systematically *heterogeneous* when it does not contain successive elliptical clauses. Then, we do not need to know the ellipsis types of elliptical clauses of this sentence. Generally, the ellipsis type determination of an elliptical clause isn't usually easy to implement. That's why, we propose a heuristic, which is clearly easier to implement. This heuristic is based on the behaviour of connectors: when some connectors of the Arabic language are successive in a sentence, the clauses which follow them have usually the same type of ellipsis (if they are elliptical). To represent this heuristic, we propose to use a matrix M[k,k] on the set {0,1} where k is the cardinality of the connector set **C**. It is called *the Coherence Matrix of Clause Connectors* and it is defined as follows:

Let c_i , $c_j \in C$.

$M[c_i,c_j] = 1$ means that the clause, which follows c_i if it is elliptical, contains the same type as the clause which follows c_j in a given sentence where these two clauses are successive.

$M[c_i,c_j] = 0$ otherwise.

M	و	ف	ثم	او	ام	ان	وان
و	0	1	1	1	1	0	0
ف	1	0	1	1	1	0	0
ثم	0	0	1	1	1	0	0
او	0	0	0	1	0	0	0
ام	0	0	0	0	1	0	0
ان	0	0	0	0	0	0	1
وان	0	0	0	0	0	0	0

Fig. 1. A reduced representation of the coherence matrix of connectors.

The use of the Coherence Matrix of Clause Connectors and of the detection method result enables us to divide the set of connectors of a given elliptical sentence in partitions. The number of this partitions determine the membership class of the elliptical sentence. If the number of partitions is equal to 1 then the elliptical sentence is homogeneous. If the number of partitions is equal to the number of clauses of the remaining clause list then the elliptical sentence is heterogeneous. Otherwise, the elliptical sentence is mixed.

The following diagram represents the different modules of the elliptical sentence location and classification.

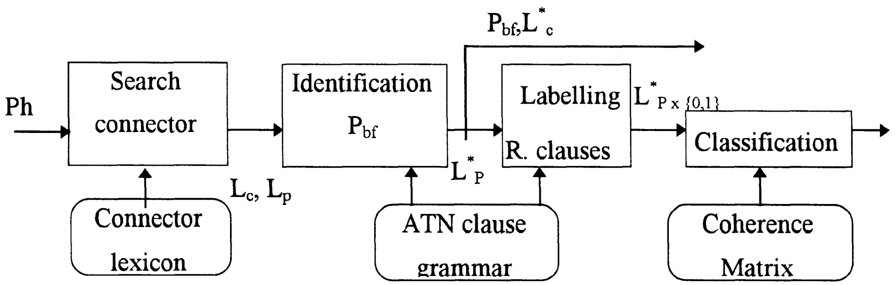


Fig. 2. The different stages of elliptical sentence treatement.

6 Conclusion

In this paper, we have presented an ATN clause grammar for the Arabic language or rather a clause parser to identify well-formed clauses in an elliptical sentence and to locate the elliptical ones. Then, we have argued why typology of ellipsis in Arabic language has almost the same characteristics as in other natural languages. Then, we have proposed a formal characterisation of the ellipsis phenomenon and a location process of elliptical sentence parts. Moreover, we have established classification criteria based on the syntactic structure of the elliptical sentences to define appropriate algorithms for ellipsis resolution.

The clause parser, the detection algorithm and the classification algorithm are implemented with C^{++} within the framework of CORTEXA system [2].

Finally, our future works concern the improvement of the used clause grammar and of the elaborated algorithms of complex elliptical sentence resolution, the resolution of some ambiguities and the effect of other phenomena (i.e., anaphora) on ellipsis resolution for the Arabic language.

References

1. Belguith, L., Ben Hamadou, A.: Marquage morpho-syntaxique robuste de mots ambigus écrit en Arabe non-voyellé. Forum de recherche en Informatique (FRI'96) Tunis (1996).
2. Ben Hamadou, A.: Vérification et correction automatique par analyse affixale des textes écrits en langage naturel: le cas de l'arabe non voyellé. Thése d'Etat en informatique, Faculté des Sciences de Tunis (1993).
3. Boguraev, B.K.: Recognizing conjunctions without the ATN framework. Automatic Natural Language Parsing. Edn. Ellis Horwood (1983).
4. Dalrymple, M., Shieber, M.S., Pereira, F.: Ellipsis and higher-order unification. Linguistic and Philosophy, 14 (1991) 399-452.
5. Gardent, C.: A Multi-Level Approach to Gapping. In the Proc. of the Stuttgart Ellipsis Workshop Bericht Nr. 29 (1992).

6. Haddar, K., Ben Hamadou, A.: Formal Description of Ellipses in Arabic Language and Resolution Process. In the Proc. of IEEE ICIPS'97, Pekin 29-31 October (1997) 1785-1790.
7. Hardt, D.: Verb phrase ellipsis: form, meaning, and processing. A dissertation in computer and information science, University of Pennsylvania (1993).
8. Huang, X.: Dealing with conjunctions in Machine Translation Environment. In Proc. of 10th International Conference on Computational Linguistics, Standford University, Palo Alto, California, July, (1984).
9. Jensen, K., Heidorn, G.E., Richardson, S.D.: Naturel Language Processing: The PLNLP Approach. Edn. Kulwer academic publishers (1993).
10. Lappin, S.: The Syntactic Basis of VP Ellipsis Resolution. In the Proc. of the Stuttgart Ellipsis Workshop Bericht Nr. 29 (1992).
11. Lappin, S.: Computational Approaches to Ellipsis Resolution. In the Proc. of the 7th Conference of the European Chapter of the Association for Computational Linguistics, Ireland, 27-31 March (1995).
12. Mast, M., Kummert, F. et al: A speech understanding and dialog system with a homogeneous linguistic knowledge base. IEEE Transactions on pattern analysis and machine intelligence, vol. 16 no. 2 (1994) 179-193.
13. Rau, F.L.: The understanding and generation of ellipses in a natural language system. Uni. of California Berkeley, Cs csd-85-227(dir), March (1985) 30 pages.
14. Sabah, G.: L'intelligence artificielle et le langage. Edn. Hermès (1989).
15. Weischedel, R.M., Sondheimer, N.K.: An improved heuristic for ellipsis processing. In the Proc. of the 20th Annual Meeting of the ACL (1982) 85-88.
16. Woods, W.: Transition networks grammars for natural language analysis. CACM 13, 10 (1970) 591-606.

A Review of Earley-Based Parser for TIG

Víctor J. Díaz, Vicente Carrillo, and Miguel Toro

Universidad de Sevilla
Departamento de Lenguajes y Sistemas Informáticos
Avda. Reina Mercedes s/n, Sevilla 41012, Spain
{vjdiaz,carrillo}@lsi.us.es

Abstract. Tree Insertion Grammar (TIG) is a compromise between Context-Free Grammars (CFG) and Tree Adjoining Grammars (TAG), that combines the efficiency of the former with the strong lexicalizing power of the latter. In this paper, we present a plain representation of TIG elementary trees that can be used directly as the input grammar for the original Earley parser without the additional considerations established in the Schabes and Waters Earley-based parser for TIG.

1 Introduction

According to Schabes, a formalism is lexicalized when each one of its basic composition structures contains some terminal symbol [6]. The lexicalization is a very interesting property from a linguistic point of view (see Abeille in [1]). In fact, most of the current linguistic theories tend to include information in the lexicon that can be considered purely syntactic. Furthermore, the time complexity of the parsers can be reduced when applying to lexicalized formalisms.

Among well-known formalisms in the literature, TAG defined by Joshi, Levy and Takahashi is a naturally lexicalized formalism [5]. The TAG formalism is a context-sensitive one, therefore, account for an important computational cost compared to CFG, see Nederhof in [7] for more details. In general, the class CFG is not naturally lexicalized, since we can define rules without terminal symbols in its right side. We can transform a CFG into another CFG' that fulfills this condition by a transformation called grammar lexicalization. In the literature, we can find different strategies of CFGs lexicalization that presents different problems [3].

An interesting formalism with respect to the problem of CFG lexicalization is Tree Insertion Grammar (TIG), presented by Schabes and Waters in [8]. TIGs are a compromise between TAGs and CFGs, and are characterized by the following: TIGs are cubic-time parsable as CFGs; TIG grammars are a subclass of TAG grammars, therefore, TIGs are naturally lexicalized. Furthermore, there are lexicalization algorithms that establish the strong equivalence of both formalisms.

A TIG is a 5-tuple (Σ, NT, I, A, S) where Σ is a set of terminal symbols, NT is a set of nonterminal symbols, I is a finite set of finite initial trees, A is a finite set of finite auxiliary trees, and S is a distinguished nonterminal symbol. The set $I \cup A$ is referred to as the set of elementary trees.

In each elementary tree, the root and interior- i.e. nonroot, nonleaf-nodes are labeled with nonterminal symbols. The nodes on the frontier are labeled with terminal symbols, nonterminal symbols or the empty string (ϵ). In initial trees, the nonterminal symbols on the frontier are marked for susbstitution. The root of at least one elementary initial tree must be labeled S. The nonterminal symbols on the frontier of an auxiliary tree are marked for susbstitution, except one nonterminal frontier node marked as the foot. The foot must be labeled with the same label as the root and the mark is normally an asterisk. The path from the root of an auxiliary tree to the foot is called the *spine*.

Auxiliary trees in which every nonempty frontier node is to the left of the foot are called *left* auxiliary trees. Similarly, auxiliary trees in which every nonempty frontier node is to the right of the foot are called *right* auxiliary trees. Other auxilary trees are called *wrapping* auxiliary trees.

Frontier nodes labeled with ϵ are referred to as empty. If all the frontier nodes of an initial tree are empty the tree is referred to as *empty*. If all the frontier nodes other than the foot of an auxiliary tree are empty, the tree is referred to as *empty*.

With respect to operations, substitution replaces a node marked for substitution with an initial tree. Adjunction replaces a node with an auxiliary tree. TIG does not allow there to be any elementary wrapping auxiliary trees or elementary empty auxiliary trees. This ensures that every elementary auxiliary tree will be uniquely either a left auxiliary tree or a right auxiliary tree.

TIG does not allow a left(right) auxiliary tree to be adjoined on any node that is on the spine of a right (left) auxiliary tree. Further, no adjunction whatever is permitted on a node μ that is to the right (left) of the spine of an elementary left(right) auxiliary tree T.

TIG allows arbitrarily many simultaneous adjunctions on a single node. Simultaneous adjunction is specified by two sequences, one of left auxiliary trees and the other of right auxiliary trees that specify the order of strings corresponding to the trees combined.

A TIG derivation starts with an initial tree rooted at S. This tree is repeatedly extended using substitution and adjunction. A derivation is complete when every frontier node in the tree(s) derived is labeled with a terminal symbol. By means of adjunction, complete derivations can be extended to bigger complete derivations.

To eliminate useless ambiguity in derivations, TIG prohibits adjunction: at nodes marked for substitution, because the same trees can be created by adjoining on the root of the trees substituted at these nodes; at foot or root nodes of auxiliary trees, because the same trees can be created by simultaneous adjunction on the nodes the auxiliary trees are adjoined on.

2 Multilayer Elementary Tree Representation

With the purpose of redefining the algorithm of Earley for TIG, Schabes and Waters [8] introduce a multilayer representation of the elementary trees of the

TIG grammars. An elementary tree will be represented by a set of CFG rules. The symbols in the left side of a rule will correspond with the direct ancestor node in an elementary tree. The right side of the rules consists of the sequence of symbols dominated in the elementary tree for the symbol on the left. We will use greek letters μ,ρ,ν to denote nodes in elementary trees and subscripts to indicate the nodes label, e.g., μ_X.

In order to specify the role performed by some nodes in an elementary tree, the set of CFG rules is enriched with the following predicates:

- $Root(\mu_X)$ when a node μ_X is the root of an initial tree
- $LAux(\mu_X)$ when a node μ_X is the root of a left auxiliary tree
- $RAux(\mu_X)$ when a node μ_X is the root of a right auxiliary tree
- $Sub(\mu_X)$ when a node μ_X is marked for substitution
- $Foot(\mu_X)$ when a node μ_X is the foot of an auxiliary tree
- $Adj(\beta, \mu_X)$ when an auxiliary tree β can be adjoined on a node μ_X

Let $G = (\Sigma, NT, I, A, S)$ be a TIG and let $a_1, \ldots, a_n$ be an input string. The Earley-style TIG parser collects states into a set called the chart. A state is a 3-tuple $[p, i, j]$, where p is a position in an elementary tree (i.e in a CFG rule in the set associated to the elementary tree) and $0 \leq i \leq j \leq n$ are integers indicating a span of the input string.

The Earley parser for TIGs (see figure) can be defined using the deductive parsing notation presented by Shieber, Schabes and Pereira [9]. The inference rules (steps) associated to initialization (1), acceptance (13) and scanning (4)(5) are interpreted in the classical sense. Predictor and completor steps are redefined due to the elementary tree representation used. For each kind of operation defined in the TIG formalism is defined a predictor/completor step. Then, six inference rules are grouped in connection with left adjunction (2)(3), right adjunction (11)(12) and substitution (7)(8). These rules use the predicates above to filter those operations that are not adjusted to the formal definition of TIG.

Furthermore, a new type of rule (Subtree Traversal) is needed (9)(10) to traverse correctly the set of CFG rules associated to an elementary tree. The simultaneous adjunction is performed by the steps adjunction predictor (left and right) and an additional scanner rule that ignores the foot node (6) of the auxiliary trees.

3 Plain Elementary Tree Representation

We will introduce a representation of TAGs, presented by Díaz and Toro [2], that tries to reduce the problems presented above. First of all, we will describe an alternative notation for trees that uses a word-based representation instead of the traditional graphical representation. The notation is as follows: a stands for $a \in \Sigma$; $X(t_1 \ldots t_n)$ stands for the elementary tree having root $X \in NT$ and direct subtrees $t_1, \ldots, t_n$. When X has not children we will use the notation X instead of $X()$.

$$Init(\mu_S) \vdash [\mu_S \rightarrow \bullet\alpha, 0, 0] \tag{1}$$

$$[\mu_A \rightarrow \bullet\alpha, i, j] \wedge LAux(\rho_A) \wedge Adj(\rho_A, \mu_A) \vdash [\rho_A \rightarrow \bullet\gamma, j, j] \tag{2}$$

$$[\mu_A \rightarrow \bullet\alpha, i, j] \wedge [\rho_A \rightarrow \gamma\bullet, j, k] \wedge LAux(\rho_A) \wedge Adj(\rho_A, \mu_A) \vdash [\mu_A \rightarrow \bullet\alpha, i, k] \tag{3}$$

$$[\mu_A \rightarrow \alpha \bullet \nu_a\beta, i, j] \wedge a = a_{j+1} \vdash [\mu_A \rightarrow \alpha\nu_a \bullet \beta, i, j+1] \tag{4}$$

$$[\mu_A \rightarrow \alpha \bullet \nu_a\beta, i, j] \wedge a = \epsilon \vdash [\mu_A \rightarrow \alpha\nu_a \bullet \beta, i, j] \tag{5}$$

$$[\mu_A \rightarrow \alpha \bullet \nu_B\beta, i, j] \wedge Foot(\nu_B) \vdash [\mu_A \rightarrow \alpha\nu_B \bullet \beta, i, j] \tag{6}$$

$$[\mu_A \rightarrow \alpha \bullet \nu_B\beta, i, j] \wedge Sub(\nu_B) \wedge Init(\rho_B) \vdash [\rho_B \rightarrow \bullet\gamma, j, j] \tag{7}$$

$$[\mu_A \rightarrow \alpha \bullet \nu_B\beta, i, j] \wedge [\rho_B \rightarrow \gamma\bullet, j, k] \wedge Sub(\nu_B) \wedge Init(\rho_B) \vdash [\mu_A \rightarrow \alpha\nu_B \bullet \beta, i, k] \tag{8}$$

$$[\mu_A \rightarrow \alpha \bullet \nu_B\beta, i, j] \vdash [\nu_B \rightarrow \bullet\gamma, j, j] \tag{9}$$

$$[\mu_A \rightarrow \alpha \bullet \nu_B\beta, i, j] \wedge [\nu_B \rightarrow \gamma\bullet, j, k] \vdash [\mu_A \rightarrow \alpha\nu_B \bullet \beta, i, k] \tag{10}$$

$$[\mu_A \rightarrow \alpha\bullet, i, j] \wedge RAux(\rho_A) \wedge Adj(\rho_A, \mu_A) \vdash [\rho_A \rightarrow \bullet\gamma, j, j] \tag{11}$$

$$[\mu_A \rightarrow \alpha\bullet, i, j] \wedge [\rho_A \rightarrow \gamma\bullet, j, k] \wedge RAux(\rho_A) \wedge Adj(\rho_A, \mu_A) \vdash [\mu_A \rightarrow \alpha\bullet, i, k] \tag{12}$$

$$Init(\mu_S) \wedge [\mu_S \rightarrow \alpha\bullet, 0, n] \vdash Acceptance \tag{13}$$

Fig. 1. Earley-Based Parser for TIGs

We will transform the word representation of $X(t_1 \ldots t_n)$ in a trivially equivalent form $X_L\, t_1 \ldots t_n\, X_R$ for every nonterminal symbol X. In other words, a category symbol X splits into two new non terminal symbols, X_L and X_R, that will divide the left and right side contexts of the symbol.

For example, the plain representation of an initial tree α with the form $S(e)$, will be $S_L\, e\, S_R$ and respectively $S_L\, e\, S^*_L\, S^*_R\, e\, S_R$ for an auxiliary tree β with the form $S(e, S^*, e)$ being S^* the foot node.

In general, the representation of an auxiliary tree β will be of the form: $X_L\, r_1\, X^*_L\, X^*_R\, r_2\, X_R$ where r_1 and r_2 are sequences of symbols, being X and X^* the root and foot symbols. If we observe carefully, we can establish that $X_L\, r_1\, X^*_L$ is just the left contextual tree dominated by the root in β with respect to his foot node. Similarly, $X^*_R\, r_2\, X_R$ will be the right context.

The adjunction operation can also be divided into two sides with respect to the spine of an auxilary tree. Suppose that β is an auxiliary tree X-rooted with frontier $w_L\, X\, w_R$ being w_L and w_R sequences of symbols. Let α be an initial tree that contains a category X with frontier $r_1\, w\, r_2$, where r_1,w,r_2 are sequences of symbols and w is the string that spans the category X. When we adjunct β in α at X we will have the frontier $r_1\, w_L\, w\, w_R\, r_2$. We can see that w_L (resp. w_R) is the string that spans the left (resp. right) contextual tree dominated by X in β.

Briefly, the trees above can be represented using a plain notation as follows:

$$\alpha = S_L\, r_1\, X_L\, w\, X_R\, r_2\, S_R$$

$$\beta = X_L\, w_L\, X_L^*\, X_R^*\, w_R\, X_R$$

With this considerations, the next three CFG-based rules can be stated to translate the elementary trees:

$$S \rightarrow S_L\, r_1\, X_L\, w\, X_R\, r_2\, S_R \text{ rule for } \alpha$$

$$X_L \rightarrow w_L\, X_L^* \text{ rule for } \beta_L$$

$$X_R \rightarrow X_R^*\, w_R \text{ rule for } \beta_R$$

where S is the label of the root of α. The plain representation of β splits into two rules representing left and right contextual sides. We eliminate the reference associated to root symbols in auxiliary trees, because adjunction operation at a root and foot nodes of an auxiliary tree are equivalent.

As we said, in TIG formalism only left and right auxiliary trees exist. If the tree is a left auxiliary tree, the right side produces only an empty string ϵ. Furthermore, it is not allowed any adjunction in the right side of the tree and it can not be adjoined a right auxiliary tree in the spine. These three constraints mean that the right part β_R only generates the empty string and, then, this part is not very important in the definition of a left auxiliary tree. Also, it is not possible to adjoin a left auxiliary tree in the root and foot nodes but we can not eliminate the foot node at all because his presence is necessary in order to obtain multiple adjunctions on the same node. Really, we only need the left side of the foot node to guarantee this kind of adjunctions. With this considerations we can eliminate the right side completely. The no-adjunction is represented by an ϵ-transition. For so much, the left auxiliary trees will have the form:

$$X_L \rightarrow w_L\, X_L^*$$

Respectively, the right auxiliary trees are defined with the following rule:

$$X_R \rightarrow X_R^*\, w_R$$

The nonterminal symbols marked for substitution can not be adjoined. These symbols do not continue the general norm of being divided into two new nonterminal symbols (left and right contexts), in this way we prevent to substitute a rule associated with an auxiliary tree.

This representation presents a fundamental advantage with respect to the multilayer representation: each elementary tree is represented by a rule. This avoids the navigation through the different levels of a tree and, therefore, the definition of the parser is simplified. We will see now how all the valid operations in TIG formalism can be performed using a plain representation.

Left adjunction is equivalent to a substitution in nonterminal symbols subscripted with L. We observe that the constraints that exist on the left adjunction are obtained with this substitution mechanism. It is not allowed to substitute a rule associated with a right auxiliary tree in the symbols associated with the right context of the spine, since these symbols do not exist. Then, the adjoining constraint of an right auxiliary tree in the spine of a left auxiliary tree is ensured.

It is not allowed to substitute a rule associated with a right auxiliary tree in the the right context symbols of left auxiliary tree, since these symbols do not exist in the rule. This guarantees the adjunction constraint of an right auxiliary tree in the right side to a left auxiliary tree. It is possible to adjoin a left auxiliary tree in the foot node. It can seem that this contradict the constraint of adjunction on the foot node, however, this operation is equivalent to a multiple left adjunction on a node in a derived tree.

Right adjunction is equivalent to a substitution in nonterminal symbols subscripted with R. The same as the left adjunction, all the constraints on the right adjunction are maintained. *Simultaneous adjunction* is equivalent to a substitution in L-symbols and later in the R-symbols. With respect to *Substitution*, the rules associated with initial trees are the only ones that can be substituted in the nodes marked for substitution, since the other rules represent auxiliary trees (labeled with right or left contexts).

4 Reviewing Earley-Based Parser for TIGs

In this section we will review the steps included in the parser presented above when it is used a plain representation. After some considerations, we will observe that the obtained parser is equal to the classical Earley parser for CFG [4].

- Scanning. The inference rules (4) and (5) are maintained. The rule (6) is not necessary, since the foot node can be substituted in multiple adjunctions. This process is equivalent to the operation *scan* in the Earley parser for CFG.
- Substitution. The predicates *Sub* and *Init* are not necessary, due to the fact that the substitution operation is already filtered. Therefore, the inference rules (7) and (8) are equivalent to the operations *predictor* and *completor* in the Earley parser for CFG, respectively.
- Subtree Traversal. The navigation of the trees is not necessary and, then, is not needed this operation. The rules (9) and (10) are equal to the operations *predictor* and *completor* in the Earley parser for CFG, respectively.
- Left and Right Adjunctions. Neither the predicates *LAux* and *RAux* are necessary, since the auxiliary trees have been labeled with new nonterminal symbols, nor the predicate *Adj*. In fact, the rules (2) and (11) is equivalent to the *predictor* in the Earley parser for CFGs, and the same the rules (3) and (12) respect to the *completor*.

5 Conclusions

The representation of TIG elementary trees can be exploited in order to take advantage of classical definition of CFGs parser. When using a multilayer representation, we must include parser rules to navigate in each elementary tree and predicates to ensure the adjunction constraints. We present an alternative word-based representation that can be used directly as the input grammar for the original Earley parser for CFGs, without extra considerations.

We argue this alternative representation captures in a more suitable way the evidence of equivalence expresiveness between CFGs and TIGs. Furthermore, the two-sides interpretation of each symbol reflects the three kind of adjunctions included in TIG formalisms.

Adding new nonterminals to the grammar does not represent a problem when constructing the derived tree. We can easily register the adjuctions performed when parsing, since we have only one rule associated with an elementary tree.

References

1. Abeille, A.: Une grammaire lexicalisee d'Arbres adjoints pour le Francais: Application a l'analyse automatique. Ph. D. Universite de Paris 7, France (1991)
2. Díaz, V., Toro, M.: Parsing TAGs with Prolog. Joint Conference on Declarative Programming, AGP'97 eds. Falaschi, M., Navarro, M., and Policrit, A. Grado, Italy (1997) 359-367
3. Díaz, M., Toro, M., Carrillo, V.: Un Algoritmo de Lexicalización de CFG mediante TAGs. Procesamiento del Lenguaje Natural **19** (1996) 201-208
4. Earley, J.: An Efficient Context-Free Parsing Algorithm. ACM **13(2)** (1970) 94-102
5. Joshi, A. K., Levy, L., Takahashi, M.: Tree Adjunct Grammars. Journal of Computer and Systems Science **10** (1975) 136-163
6. Schabes, Y.: Mathematical and Computational Aspects of Lexicalized Grammars. Ph. D. University of Pennsylvania, Philadelphia, USA (1990)
7. Nederhof, M.: Solving the Correct-Prefix Property for TAGs. 5th Meeting of Mathematics of Language. Schloss Dagstuhl, Germany (1997)
8. Schabes, Y., Waters, R.: Tree Insertion Grammars: A Cubic-Time, Parsable Formalism that Lexicalized Context-Free Grammars Without Changing the Trees Produced. Computational Linguistic **21(4)** (1995) 479-515
9. Shieber, S., Schabes, Y., Pereira, F.: Principles and Implementation of Deductive Parsing. Journal of Logic Programming, **4(1&2)** (1995) 3-36

Generating Explanations from Electronic Circuits *

Takushi Tanaka

Fukuoka Institute of Technology
3-30-1 Wajiro-Higashi, Higashi-ku, Fukuoka 811-0295, Japan

Abstract. We have developed a prototype system which explains the structure and behavior of electronic circuits to assist electrical engineers. This system analyzes the structure of the given circuit and generates explanations of its operation. In our system, circuits are viewed as sentences, and their elements as words, in a word-order-free language. Knowledge of circuit structures is coded as grammar rules. Using these grammar rules, the given circuit is parsed and explanations are generated from the result of the parse.

1 Introduction

Electronic circuits are designed by engineers as goal oriented compositions of basic circuits with specific functions. Therefore, an electronic circuit has a hierarchical structure of functional blocks with specific goals. This means understanding a circuit consists of finding this hierarchical structure of functional blocks and rediscovering the designer's original intentions. The ability to understanding circuits in this way is a prerequisite for both circuit design and troubleshooting.

As an aid for understanding circuits, we have developed a system that generates explanations from given circuits. Circuits are viewed as sentences and their elements as words. Circuit structures are defined by a DCG-like logic grammar formalism [2] called DCSG (Definite Clause Set Grammars [3]), developed for analyzing free-word-order languages. A set of grammar rules, when translated into Prolog clauses, forms a logic program that performs top-down parsing. When an unknown circuit is given, such a logic program will analyze the circuit and derive a parse tree for the circuit [4].

An extension to the DCSG formalism adds an additional field for non-syntactic information. This field is called the semantic field [4]. Using this field, we can associates structures with their meaning. In this paper, we associate specific semantic information with functional blocks. This information specifies circuit goals, conditions, behaviors, and causality relationships for voltages and currents. This information is collected through the parsing process, placed in an output stream, and then becomes the source data for generating explanations for the circuit's behavior or design.

* This work was supported in part by the Japan Ministry of Education under Grant-in-Aid for Science Research 08680418.

2 Generated Sentences

When a circuit is given, the system parses the circuit and generates explanations. The following examples are explanations generated from the circuit *cd42* in Figure 1. Since we use a simple translation mechanism from logical predicates to sentences, the resultant language is a bit awkward. Example 1 is a structural explanation for the circuit *cd42*. Each sentence is generated by direct translation from the parse tree in top-down depth-first manner. Sentences in example 2 include explanations of circuit goals, behaviors, and voltage and current causality. These are generated from the semantic information collected through the parsing process. Since sentences are printed in abbreviation mode, all names of functional blocks are abbreviated, e.g. "opAmpl(...)".

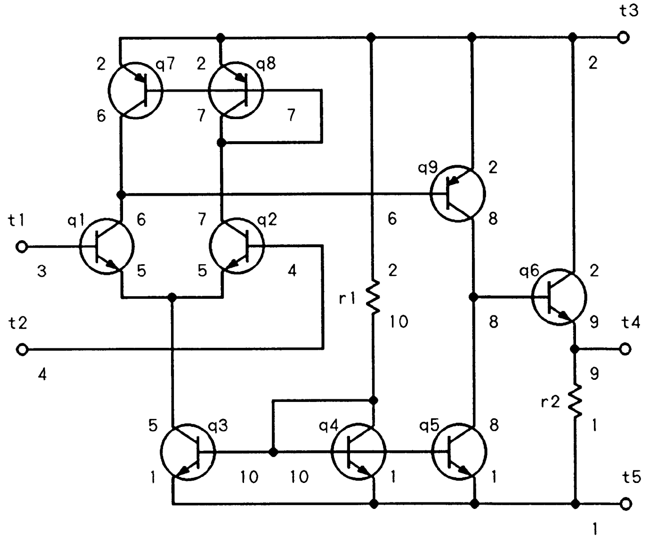


Figure 1. Operational amplifier *cd42*.

Example 1. Structural explanations

- This circuit is parsed as opAmpl(...).
- The circuit opAmpl(...) is an operational amplifier.
- It consists of sdAmp(...), pnpCE(...), and npnEF(...).
- The circuit sdAmp(...) is a single-ended differential amplifier.
- It consists of ecup(...), al(...), and sink(...).
- The circuit ecup(...) is an emitter-coupled pair.
- It consists of q1 and q2.
- The circuit al(...) is an active load.
- It consists of pdtr(...) and q7.
- The circuit pdtr(...) is a diode-connected transistor.
- It consists of q8.
- The circuit sink(...) is a current source.

- It consists of vreg(...) and q3.
- The circuit vreg(...) is a voltage regulator.
- It consists of ndtr(...) and r1.
- The circuit ndtr(...) is a diode-connected transistor.
- It consists of q4.
- . . .

Example 2. Explanations generated from semantic information

- The element t5 is a minus_Vcc_terminal.
- The element t3 is a plus_Vcc_terminal.
- The element t4 is a output_terminal.
- The element t2 is a negative_input_terminal.
- The element t1 is a positive_input_terminal.
- The functional block npnEF(...) has a goal to provide low_output_impedance.
- The functional block pnpCE(...) has a goal to provide high_voltage_gain.
- The functional block sdAmp(...) has a goal to amplify differential_inputs.
- The voltage between 8 and 1 controls the voltage between 9 and 1.
- The element r2 has a goal to sink current_from_load.
- The element q6 has a goal to amplify current.
- The functional block npnEF(...) has a function to convert impedance.
- If q9 is active, then the current from q9 to 8 is equal to the product of the hfe and the current from q9 to 6.
- The current from q9 to 6 controls the current from q9 to 8.
- The functional block sink(...) has a goal to provide high_impedance_load.
- The element q9 has a goal to amplify current.
- The voltage between 10 and 1 controls the current from 8 to q5.
- The functional block sink(...) has a function to generate constant_current.
- If the current from 10 to ndtr(...) is greater than 0, then the voltage between 10 and 1 is equal to Vbe.
- The functional block vreg(...) has a function to generate regulated_voltage.
- The functional block ndtr(...) works as a diode.
- The functional block sink(...) has a goal to suppress common_mode_gain.
- The functional block al(...) has a goal to duplicate current_gain.
- . . .

3 Circuit Grammar with Semantic Field

In designing electronic circuits, engineers use many basic circuits with specific functions, and view these circuits as macro elements or macro devices. Functional blocks such as "diode-connected transistor", "voltage regulator", "current source", "active load", and "differential amplifier" are found in the circuit *cd42* (Figure 1). The circuit *cd42* itself can also be viewed as an "operational amplifier" functional block. These functional blocks are defined as non-terminal symbols. We will present several sample grammar rules. The grammar rules are discussed in detail in an earlier paper [4].

The simplest functional block, the "diode-connected transistor", is defined by (1) as a non-terminal symbol (Figure 2). All terminal symbols in grammar rules are surrounded by square bracket. The first line of rule (1) means that an NPN-transistor Q with the base and the collector connected to the same node "A" functions as a diode. As is the case with terminal symbols, the first argument of non-terminal symbols is the circuit name, and other arguments are the connected nodes in the circuit. Namely, "$ndtr(Q)$" is a name given to the diode-connected transistor, and "A" and "C" are connecting nodes. The additional field in curly brackets is a semantic field containing information other than circuit structures. This field in rule (1) indicates that the circuit $ndtr(Q)$ functions as a diode. This field is used in parsing context-dependent circuits as discussed in an earlier paper [4].

$$\begin{aligned} dtr(ndtr(Q), A, C) \longrightarrow\ & [npnTr(Q, A, C, A)], \\ & \{func\_as(ndtr(Q), diode)\}. \end{aligned} \tag{1}$$

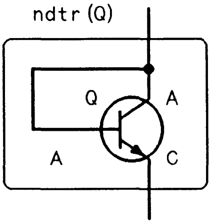


Figure 2. Diode-connected transistor.

The following grammar rule defines the non-terminal symbol *Vbe-voltage regulator* which has Vbe (0.6 - 0.7 volts) as its output. The first two lines define the circuit structure shown in Figure 3. The other lines in curly brackets define its functions and conditions. That is, the circuit $vreg(D, R)$ has a function that outputs a regulated_voltage, and if a current flows from node Out to device D, its output voltage becomes vbe.

$$\begin{aligned} vbeReg(vreg(D, R), Vp, Com, Out) \longrightarrow\ & dtr(D, Out, Com), \\ & res(R, Vp, Out), \\ \{func(vreg(D, R), generate(regulated\_voltage)), & \\ cond(gt(current(Out, D), 0), eq(voltage(Out, Com), vbe))\}. & \end{aligned} \tag{2}$$

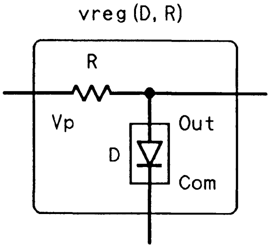


Figure 3. Vbe-voltage regulator.

A simple current source "*sink*(VR, Q)" is defined by the following grammar rule (3). The current source consists of a Vbe-voltage regulator and an NPN-transistor (Figure 4). The semantic field shows its function and causality. That is, the circuit $sink(VR, Q)$ has the function of $generate(constant\_current)$, and its $current(In, Q)$ is controlled by the $voltage(B, Com)$.

$$\begin{aligned} cSink(sink(VR,Q), In, Com) \longrightarrow\ & vbeReg(VR, \_, Com, B),\\ & [npnTr(Q, B, Com, In)],\\ \{func(sink(VR,Q), & generate(constant\_current)),\\ ctrl(voltage(B, Com), & current(In, Q))\}. \end{aligned} \tag{3}$$

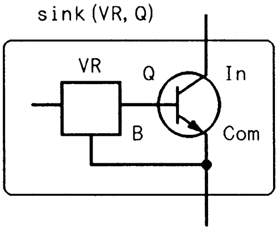


Figure 4. Simple current source (sink-type).

An active load "$al(D, Q)$" is defined by the grammar rule (4). The structure shown in Figure 5 is defined by the first two lines. The semantic field defines its function, causality, and electrical condition as: The circuit $al(D, Q)$ has the function of mirroring current. The reference current from the circuit $al(D, Q)$ to node *Ref* controls the output current from transistor Q to node Ld. On the condition of flowing reference current, the output current becomes equal to the reference current.

$$\begin{aligned} activeLoad(al(D,Q), Ref, Vp, Ld) \longrightarrow\ & dtr(D, Vp, Ref),\\ & [pnpTr(Q, Ref, Vp, Ld)],\\ \{func(al(D,Q), & mirror(current)),\\ ctrl(current(al(D,Q), Ref), & current(Q, Ld)),\\ cond(gt(current(al(D,Q), Ref), 0), & \\ eq(current(Q, Ld), & current(al(D,Q), Ref)))\}. \end{aligned} \tag{4}$$

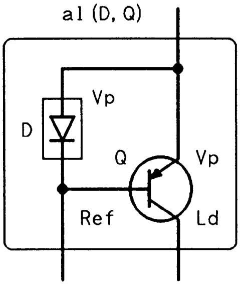


Figure 5. Active load (source type).

The single ended differential amplifier is defined using functional blocks "emitter-coupled pair", "active load", and "current source" by the following rule (5). The structure of an operational amplifier is also defined using "single-ended differential amplifier", "common emitter", and "emitter follower".

$$\begin{aligned} & sDiffAmp(sdAmp(EC, AL, CS), B1, B2, C1, Vp, Vm) \longrightarrow \\ & \qquad eCoupledPair(EC, B1, B2, E, C1, C2), \\ & \qquad activeLoad(AL, C2, Vp, C1), \\ & \qquad cSink(CS, E, Vm), \\ & \quad \{func(sdAmp(EC, AL, CS), amplify(differential\_inputs)), \\ & \quad goal(EC, provide(differential\_inputs)), \\ & \quad goal(AL, duplicate(gain)), \\ & \quad goal(CS, suppress(common\_mode\_gain))\}. \end{aligned} \tag{5}$$

4 Extended DCSG

Grammar rules for circuit structures are translated into Prolog clauses by the DCSG translation procedure [3]. The general form of the translation from grammar rules to definite clauses is shown by formulas (6) and (6)'.

$$A \longrightarrow B_1, B_2, ..., B_n, \{C_1, C_2, ..., C_m\}. \tag{6}$$

$$\begin{aligned} ss(A, S_0, S_n, E_0, E_{n+1}) :- \ & ss(B_1, S_0, S_1, E_0, E_1), \\ & ss(B_2, S_1, S_2, E_1, E_2), \\ & \ldots, \\ & ss(B_n, S_{n-1}, S_n, E_{n-1}, E_n), \\ & E_{n+1} = [C_1, C_2, ..., C_m | E_n]. \end{aligned} \qquad (6)'$$

Here, all of the symbols $B_1, ..., B_n$ in the grammar rule (6) are assumed to be non-terminal symbols. When the clause (6)' is used in parsing, an object circuit (free-word-order sentence) is substituted into S_0, and the functional block "A" (non-terminal symbol) is identified as a subset of S_0. Procedurally, the clause (6)' can be read as: In order to show A to be a subset of S_0, show B_1 to be a subset of S_0, show B_2 to be a subset of S_1 which is the remainder of B_1 from S_0, ... , show B_n to be a subset of S_{n-1}, and set the remainder to S_n. That is, the stream of $S_0, S_1, ..., S_n$ in a parsing process contains the unknown part of the circuit, and gradually decreases as the parsing process proceeds. If the whole object circuit is identified as the functional block "A" and no unknown part remains in the object circuit, then S_n becomes empty.

Grammar rules that generate terminal symbols differ slightly. The following grammar rule, which generates a terminal symbol (7), is translated into (7)'.

$$B \longrightarrow [D]. \tag{7}$$
$$ss(B, S_0, S_1, E_0, E_0) \longrightarrow member(D, S_0, S_1). \qquad (7)'$$

The predicate member is defined as:

$$\begin{array}{l} member(X,[X|R],R). \\ member(X,[U|Y],[U|R]) :- member(X,Y,R). \end{array} \tag{8}$$

The clause (7)' can be read as: In order to show "B" to be a subset of S_0, show D to be a member of S_0. "B" is the name of subset which contains one element "D".

The sequence $E_0, E_1, ..., E_n$ is used to transfer semantic information other than syntactic structures through parsing. When the grammar rule (6) is applied in parsing, namely a functional block "A" is found in the given circuit, the semantic information "$C_1, C_2, ..., C_m$" for the functional block "A" is added into the stream. Each time a functional block is found, the stream of semantic information increases. When the whole circuit is parsed successfully, the variable S_n becomes empty, while the variable E_n contains the semantic information items for the circuit.

5 Generating Explanations

In top-down parsing of free-word-order languages, grammar rules are applied to a starting symbol until terminal symbols are generated, and those terminal symbols are found in the object sentence. When all of the terminal symbols are found, and no unknown terminal symbols remain in the sentence, then the top-down parsing succeeds. If a given circuit is a grammatical one, the following goal clause (9) successfully parses the circuit. Although we have not introduced a single starting symbol, the variable S in the following goal clause works as the starting symbol which generates all of the non-terminal symbols. Since the third argument of "ss" is a null circuit, the goal clause asks whether the whole circuit in $ObjectCircuit$ is identified as a non-terminal symbol S.

$$?-\ \ ss(S, ObjectCircuit, [\,], \_, \_). \tag{9}$$

When the following goal clause successfully parses the whole circuit of "$cd42$" in Figure 1, we acquire the values of variables in the goal clause. The first line binds the circuit topology of $cd42$ to the variable CT, then next line identifies the functional block S in the circuit CT.

$$\begin{array}{l} ?-\ \ cd42(CT), \\ \qquad ss(S, CT, [\,], [\,], E). \end{array} \tag{10}$$

$$\begin{array}{l} S = opAmp(opAmp1(sdAmp(ecup(q1,q2), \\ \qquad\qquad\qquad\qquad\qquad al(pdtr(q8),q7), \\ \qquad\qquad\qquad\qquad\qquad sink(vreg(ndtr(q4),r1),q3)), \\ \qquad\qquad\qquad pnpCE(q9, sink(vreg(ndtr(q4),r1),q5)), \\ \qquad\qquad\qquad npnEF(q6,r2)), \\ \qquad\qquad\qquad 3,4,9,2,1) \end{array}$$

$$
\begin{aligned}
E = [&isa(t5, minus\_Vcc\_terminal),\\
&isa(t3, plus\_Vcc\_terminal),\\
&isa(t4, output\_terminal),\\
&isa(t2, negative\_input\_terminal),\\
&isa(t1, positive\_input\_terminal),\\
&goal(npnEF(q6, r2), provide(low\_output\_impedance)),\\
&goal(pnpCE(q9, sink(vreg(ndtr(q4), r1), q5)),\\
&\qquad provide(high\_voltage\_gain)),\\
&goal(sdAmp(ecup(q1, q2), al(pdtr(q8), q7),\\
&\qquad\qquad\qquad sink(vreg(ndtr(q4), r1), q3)),\\
&\qquad amplify(differential\_inputs)),\\
&ctrl(voltage(8, 1), voltage(9, 1)),\\
&goal(r2, sink(current\_from\_load)),\\
&goal(q6, amplify(current)),\\
&func(npnEF(q6, r2), convert(impedance)),\\
&cond(state(q9, active), eq(current(q9, 8), mlt(hfe, current(q9, 6)))),\\
&ctrl(current(q9, 6), current(q9, 8)),\\
&goal(sink(vreg(ndtr(q4), r1), q5), provide(high_impedance_load)),\\
&goal(q9, amplify(current)),\\
&func(pnpCE(q9, sink(vreg(ndtr(q4), r1), q5)),\\
&\qquad provide(high_voltage_gain)),\\
&ctrl(voltage(10, 1), current(8, q5)),\\
&func(sink(vreg(ndtr(q4), r1), q5), generate(constant_current)),\\
&cond(gt(current(10, ndtr(q4)), 0), eq(voltage(10, 1), vbe)),\\
&\ldots])
\end{aligned}
$$

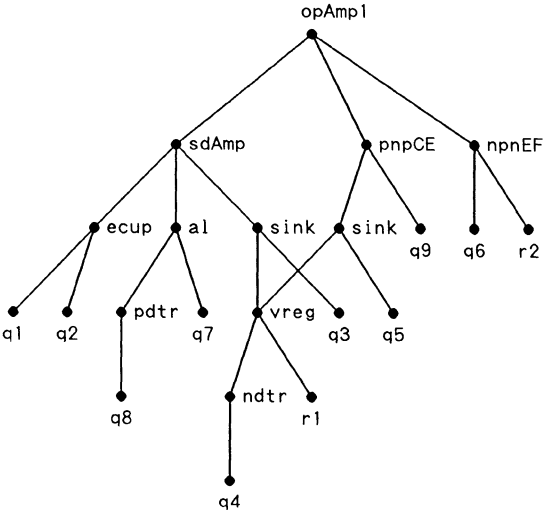


Figure 6. Parse tree

The first argument of the value S is the name given to the parsed circuit; the others are nodes connected to the external terminals. This name records the successful goals during parsing and forms a hierarchical structure of functional blocks that can be viewed as a parse tree for the circuit (Figure 6).

The circuit structure explanations shown in *Example 1* (Section 2) are generated by direct translation of this parse tree. This can be done by simply executing the parse tree as a goal clause. Sentence patterns are defined for each name of functional block which becomes a node of parse tree. For example, the name $sdAmp(X, Y, Z)$ has two sentence patterns as shown in (11), and generates the fourth and the fifth sentences in *Example 1*. The term "$ab(X)$" changes "X" into abbreviation form. The predicate "$m\_write(X)$" writes a sentence according to the list X. That is, atoms in X are printed as it is and lists in X are executed as subgoals. The last line "X, Y, Z" of the definition makes subgoals for top-down depth-first explanations.

$$\begin{aligned}&sdAmp(X, Y, Z) :-\\ &\quad m\_write(['The\ circuit\ sdAmp(...)\ is\ a\ single\text{-}ended\ differential\ amplifier']),\\ &\quad m\_write(['It\ consists\ of\ ', ab(X), ',\ ', ab(Y), ',\ and\ ', ab(Z)]),\\ &\quad X, Y, Z.\end{aligned} \tag{11}$$

The variable "E" in goal (10) contains the semantic information collected during parsing. The explanations of circuit behavior in *Example 2* are generated from this semantic information, which includes circuit goals, functions, control, and conditions. We use the same technique to generate sentences here as we did in *Example 1*, that is, executing collected terms as goal clauses. Sentence patterns are defined for each term, e.g. "*goal*", "$func$", "*ctrl*", and "*cond*". Logical sentences in the variable "E" are translated into the explanations in *Example 2* using these sentence patterns. For example, the following clause (12) is defined for "*goal*".

$$goal(X, Y) :- m\_write([ar(X), '\ has\ a\ goal\ ', to(Y)]). \tag{12}$$

The term "$ar(X)$" adds proper header such as article and change X to proper form. The phrase "The functional block npnEF(...)" in the 6th sentence of *Example 2* is generated by this "$ar(X)$". The term "$to(Y)$" generates an infinitive clause from "Y" such as "to provide low_output_impedance" in the same sentence.

The following clauses are defined for electrical conditions, and used to generate 20th sentence in *Example 2*. The predicate "m_write2" is almost same as "m_write" but does not output a period and a line-feed.

$$cond(X, Y) :- m\_write(['If\ ', X, ',\ then\ ', Y]). \tag{13}$$
$$gt(X, Y) :- m\_write2(['the\ ', X, '\ is\ greater\ than\ the\ ', Y]). \tag{14}$$

6 Conclusions

We have developed a mechanism for generating explanations of the structure and behavior of electronic circuits. Circuits are viewed as sentences generated by circuit grammars. Knowledge of circuit structures is coded as grammar rules. Since each grammar rule defines a functional block, knowledge concerning the function of the functional block is added into the rule as additional semantic information. Both the parse tree and the semantic information collected through the parsing process become sources for generating explanations of circuit structure and behavior.

Through experiments in generating explanations, we have found that this approach generates too many explanations for the given circuit, and that the order defined by the parsing process is not appropriate. That is, users do not need all the explanations that are generated, and may need the explanations in a different order than the order currently produced. We feel that mechanisms for interactive selection and natural ordering of the explanations will be necessary, since different users will have different interests. For example, one user may be interested in the overall purpose (goal) of the whole circuit, while another user may be interested in the functions of specific elements, or causality relations between voltages and currents, Also, more work is required in generating natural English explanations both in sentences and in sequences of explanations. In particular, we hope to apply earlier work on discourse generation [1] to this problem. We are now developing a user interface for interactive explanation using the circuit editor.

7 Acknowledgments

I would like to thank Prof. Paul Tarau for the BinProlog system used these experiments, and Mr. David Littleboy for helpful advice on both content and style.

References

1. Eduard H. Hovy, *Automated discourse generation using discourse structure relations*, Artificial Intell,. Vol.63, pp. 341–385, 1993.
2. F.C.N. Pereira and D.H.D. Warren, *Definite Clause Grammars for Language Analysis*, Artificial Intell., Vol.13, pp. 231–278, 1980.
3. Takushi Tanaka, *Definite Clause Set Grammars: A Formalism for Problem Solving*, J. Logic Programming, Vol.10, pp. 1–17, 1991.
4. Takushi Tanaka, *Parsing Circuit Topology in A Logic Grammar*, IEEE-Trans. Knowledge and Data Eng., Vol.5, No.2, pp. 225–239, 1993.
5. Paul Tarau, *BinProlog 5.75 User Guide*, Dept. d'Informatique, Univ. de Moncton, 1997.
6. *101 Analog IC Designs*, Sunnyvale, CA: Interdesign Inc., 1976.

The Effect of a Dynamical Layer in Neural Network Prediction of Biomass in a Fermentation Process

Majeed Soufian, Mustapha Soufian & M. J. Dempsey*

Mechanical Engineering, Design and Manufacture, Biological Sciences,*
Manchester Metropolitan University, Manchester, M1 5GD, UK

Abstract. In this paper, computational intelligence has been considered as a tool (software sensor) for state-estimation and prediction of biomass concentration in a simulated fermentation process. Two different paradigms of an artificial neural networks have been introduced as possible computational engines. Inclusion of process dynamics is inherent within the second paradigm, as a pre-processing layer. The constructed computational engines 'infer' the production of biomass from easily measured on-line variables. First and second-order non-linear optimisation methods are used to train the neural networks. It is shown that the use of the pre-processing layer which contains dynamical elements, produces better results and shows significant improvement in the convergence rate of the neural networks.

1. Introduction

The productive use of micro-organisms in the biological process industries relies upon suitable control strategies to steer the microbe's physiology towards commercial ends. This requires an understanding of microbial physiology in order to design an appropriate controller, and suitable sensors to provide the state estimates. A common misconception amongst non-microbiologists is that microbes are simple and primitive, this is not true. Since life evolved about 3·5 x 10^9 years ago [1], the twin forces of mutation and natural selection have resulted in advanced regulatory processes for cellular metabolism. Although microbes may look simple, they posses sophisticated sensor systems which are linked to intricate, multi-level strategies for control of their metabolism [2].

Despite advances in our knowledge of cellular processes, we do not yet have sufficient understanding for mastery of fermentation performance through externally imposed control. Furthermore, economic operation is further hampered by the difficulty in modelling these highly nonlinear systems using conventional mathematical approaches, and the restricted range of sensors for state estimation. As a result, these processes are controlled in open loop manner; which can lead to sub-optimal production because, for example, nutrients and energy are used in excess of the minimum actually required. The advent of computational intelligence techniques, such as neural net computing, fuzzy logic and genetic algorithms, has led to improved modelling [3]; whilst the development of 'software sensors' [4,5] allows the on-line estimation of parameters such as product and biomass concentrations, which normally have to be measured off-line.

Many attempts have been made to use linear and non-linear system identification techniques to construct mathematical models as ‘software sensors’ to infer, for example, biomass and other product concentrations. The term software sensor is a name for tools which use mathematical models and algorithms with available on-line data to provide adequate information about the present and future state of the key process parameters. The use of conventional linear and non-linear system identification techniques to construct mathematical models for fermentation processes has had limited success [6].

Artificial Neural Network (ANN) paradigms are universal approximators [7] and are capable of approximating general non-linear mappings to any desired degree of accuracy. It has been widely shown that these techniques give better results in modelling and identification of non-linear and unknown system dynamics in comparison to conventional linear and non-linear system identification algorithms [8,9]. In this paper, Two different neural networks were designed as a software sensor for state-estimation and prediction of biomass concentration in a simulated fermentation process [10]. Initially, first and second-order non-linear optimisation methods were used to train neural networks. Subsequently, a pre-processing layer containing dynamic elements was added to the neural network architecture. The network was then trained using a second order, nonlinear optimisation method. Results of the identification methods are presented and compared. It was found that neural networks containing dynamic pre-processing elements produced better results and showed significant improvement in the convergence rate.

2. Artificial Neural Networks

An ANN is a parallel, distributed, information processing structure consisting of processing elements interconnected via unidirectional signal channels [11]. Every ANN has an input layer, an output layer and some optional layers between input and output layers, which are called hidden layers. In each layer, there are several processing elements and each processing element has a single output connection that branches into as many collateral connections as desired to the next layer. Figure 1 shows a feed-forward ANN architecture with two hidden layers. The connections amongst processing elements are appropriately weighted to obtain a desired output from the ANN. There is a possibility for an additional bias term or offset for each processing element. The processing elements in each layer are termed as neurones. The number of neurones in the input and output layers can be determined by the number of input and output variables in the physical system. The number of hidden layers and the number of neurones in the hidden layers are arbitrary and can vary from zero to any finite number. The processing element output signal can be of any mathematical type desired. For each layer, it can be a different linear or non-linear function. However, for obtaining a non-linear mapping, a non-linear function is often used in the hidden layer and a linear function in the output layer for scaling purposes. For a logistic sigmoid, which is the most common non-linear function for ANN, the output signal a_i^λ of processing element 'i' in the layer λ can be written as:

$$a_i^\lambda = f_i^\lambda(n_i^\lambda) = \frac{1}{1+e^{-n_i^\lambda}} \tag{1}$$

where

$$n_i^\lambda = \sum_{j=1}^{s^{\lambda-1}} w_{ij}^\lambda f_j^{\lambda-1} + b_i^\lambda \tag{2}$$

and w_{ij}^λ 's are associated connection weights among all $s^{\lambda-1}$ neurones in layer λ-1 and neurone number 'i' in layer λ, and b_i^λ is the bias for the related neurone. The Taylor series expansion of equation (1) given by

$$f_i^\lambda(n_i^\lambda) = c_0 + c_1 n_i^\lambda + c_2(n_i^\lambda)^2 + c_3(n_i^\lambda)^3 + \cdots \tag{3}$$

indicates that the neuron operation can be considered as an infinite series of non-linear expansion of its inputs.

3. Inclusion of Dynamics in Neural Networks

Although it has been shown that ANNs are capable of approximating general non-linear mappings to any desired degree of accuracy, these are limited to non-linear mapping, and dynamics are not inherently included within their architectures. In many practical applications (such as fermentation processes) the transient period, which cannot be described by static input to output mapping (as in the case of steady-state), is the main focus of processes. This makes identification of dynamical processes using ANNs difficult. To solve this difficulty, the fully Recurrent ANN (RANN) approach of Williams et al. [12] can be considered. However, in practice, existing sensors do not allow biomass concentration to be measured on-line, accurately and frequently. Off-line measurements are only available with a significant time delay, which in the case of biomass measurements can extend beyond the termination of the fermentation [6]. This make the use of RANN impossible for this application.

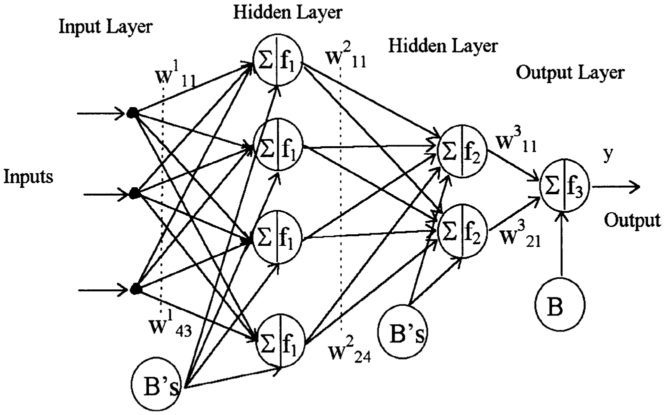


Fig. 1. A Static Feed-Forward ANN with two hidden layers

For fermentation processes, another approach is suggested, based on integrating time operation inside the ANN architecture. Klopf [13], Klopf et al [14], Sutton and Barto [15] considered the role of time in natural intelligence. Also, Holden [16] suggested that biological neural networks in the brain have dynamical elements. Therefore, it seems natural to incorporate time in the ANN operation. Although no real mathematical analysis is presented here to prove the efficiency of the purposed architecture, experimental results showed improvements. The suggested ANN structure contains linear dynamical elements $g_i^\lambda(D)$ defined as a first order transfer operation:

$$g_i^\lambda(D) = \frac{1}{D + b_i^\lambda} \tag{4}$$

where b_i^λ play the role of an inverse time constant in a first order system.

3.1. Behaviour of the Dynamical Neural Networks

More insight into the ANN with a dynamical layer can be obtained by expressing the linear dynamical element (4) in the state space form:

$$\begin{aligned} \dot{x}_i^\lambda(t) &= A_i^\lambda x_i^\lambda(t) + B_i^\lambda u_i^\lambda \\ y_i^\lambda(t) &= C x_i^\lambda(t) + D u_i^\lambda \end{aligned} \tag{5}$$

where u_i^λ and y_i^λ are input and output of the neuron in the layer λ respectively. The A_i^λ is equal to $-b_i^\lambda$ and for simplicity assume $C = B = 1$ and $D = 0$, i.e. there is no bypass connections. Now depending on the position of dynamical elements the behaviour of the overall network can be predicted. For example if linear dynamical elements are located in the last hidden layer ($\lambda = \Lambda-1$), and if linear activation functions are considered for the output layer then output of the network is a linear combination of each dynamical element so the dynamics of the ANN can be describe as follow:

$$\dot{x}_i^{\Lambda-1}(t) = A_i^{\Lambda-1} x_i^{\Lambda-1}(t) + \sum_j^{s^{\Lambda-1}} w_{ij}^{\Lambda-1} f_i^{\Lambda-2} \tag{6}$$

where f is the nonlinear activation function defined in equation 1. The output of ANN can be written as:

$$y_i^\Lambda(t) = \sum_{j=1}^{s^{\Lambda-1}} w_{ij}^\Lambda x_i^{\Lambda-1}(t) + b_i^\Lambda \tag{7}$$

From equations 6 and 7 it can be concluded that ANN presented by the above architecture, represents systems with nonlinear elements in their inputs and the role of static part of ANN is to approximate these nonlinearities. A majority of the process industries can be defined in a way that the gain of the process are dependent on operating point and the above ANN structure naturally matches them.

Another purposed topology is shown in Figure 2, where the dynamical element is located in the first hidden layer (i.e., λ=1), between the summation elements and nonlinear functions. It is possible to locate them as a separate hidden layer. In this case, the dynamics of the ANN can be obtained as:

$$\dot{x}_i^1(t) = A_i^1 x_i^1(t) + \sum_{j}^{s^{\Lambda-1}} w_{ij}^1 a_i^0 \tag{8}$$

where a_i^0 represents the ith input of ANN. The overall map between the output of dynamical layer and the output of ANN is shown by a nonlinear function *h*:

$$y_i^{\Lambda} = h(a^{\lambda}, ..., a^1, x^1(t)) . \tag{9}$$

which indicate that this architecture is useful for processes where nonlinearities located in their outputs. From these two topology, it can be concluded that if the linear dynamical element is located between the two hidden layers, the ANN will have the behaviour of a dynamical system which the nonlinearities occur in input and output of the system

3.2. Training Algorithms

The outputs of any ANN are dependent on weighting on the constituent neurones. The determination of these weights is essential for the ANN to learn the output response of target system. The process of finding the weights and biases, which produces the best match between the desired map and the output of the ANN, is called the training process. In a training process, all weights are modified in such a manner to minimises an error function, which is formalised by the normalising the summation of squared errors $\varepsilon(I) = y(i) - a^{\Lambda}(i)$ between the outputs of the ANN (a^{Λ}) with s^{Λ} neurones at output layer Λ and a training output set Y:

$$\min_{W \in R^Q} \text{imise}\, J = \frac{1}{2k}\left\| Y - A^{\Lambda} \right\|_2^2 = \frac{1}{2N}\sum_{i=1}^{N}\sum_{j=1}^{s^{\Lambda}} [y_{i,j} - a_{i,j}^{\Lambda}]^2 \tag{10}$$

where A^{Λ} is the N consecutive output of ANN and Q is the number of weights in the network. A training set is a set of N input/output pairs of the desired mapping, which

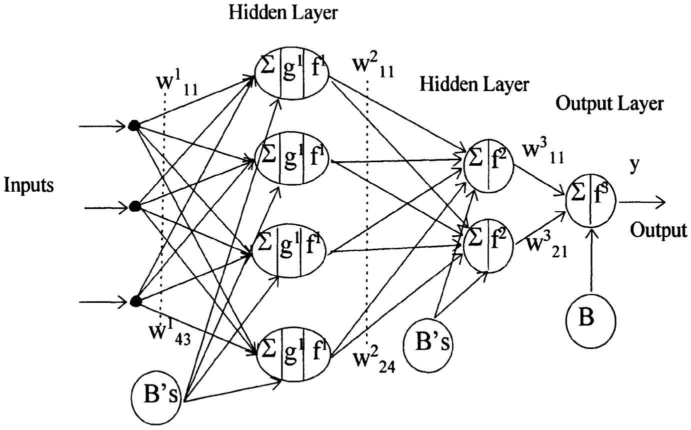


Fig. 2. A Feed-Forward ANN with Dynamical elements in the first hidden layer.

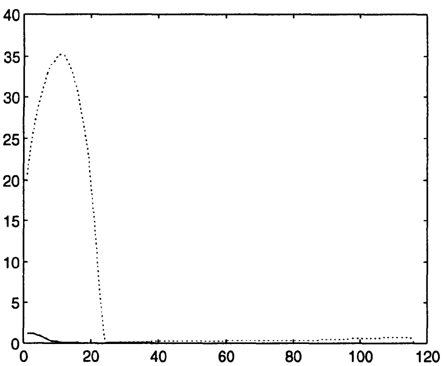


Fig.3. Two substrates s1 (dotted line) and s2 (solid line) are used for feeding micro-organism.

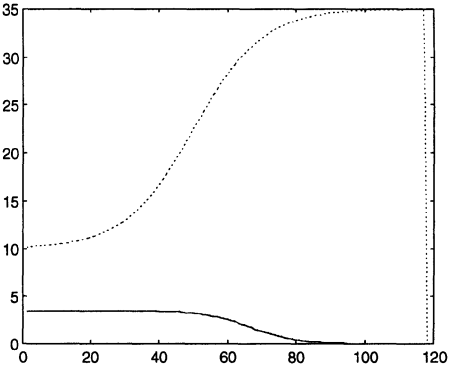


Fig.4. Two feeds, f_1 (dotted line) and f_2(solid line) are used to provide the substrates.

are supposed to be learned by the ANN. The error function is a multivariate measure that depends on weights, so that the training process is basically an unconstrained non-linear optimisation problem. A quadratic objective function is the simplest non-linear function. It is expected that any optimisation technique that minimises a general non-linear function successfully works well on a quadratic one. Besides, any non-linear function near its optimum behaves approximately as a quadratic function. There are several well known methods to search the optimal value of an objective function in optimisation theory, which can be applied to train the neural networks. The first order and Levenberg-Marquardt second-order non-linear optimisation methods [17,18,] were used here to train the ANN.

4. Experimental Results

The target system in this study is a black box plant simulator of a *fed-batch fermentation* process used in a fed-batch fermentation process modelling and control competition [10]. It produce a secondary metabolite as the product. The micro-organism in this process need two substrates, s_1 and s_2, for growth and production (Figure 3). Two feeds, f_1 and f_2, are used to provide these substrates to the culture (Figure 4). The aim was to study the effects of available measurements (s_1, s_2, f_1 , f_2) in the productivity of biomass. For this purpose, other process environment variables, such as temperature and pH, were assumed to be kept constant at their optimum level.

Before starting the experiments, the practical constraints were considered and simulation data were used as experimental data i.e. the unavailability of regular and accurate measurement of biomass was considered and therefore it was not used as input data to the ANN, all available data were normalised and white noise added to inputs.

Two different networks were designed for the non-linear identification of the target system. Initially, a static ANN (without any dynamical elements) were trained with available measurements (s_1, s_2, f_1, f_2) using first and second-order non-linear optimisation methods.

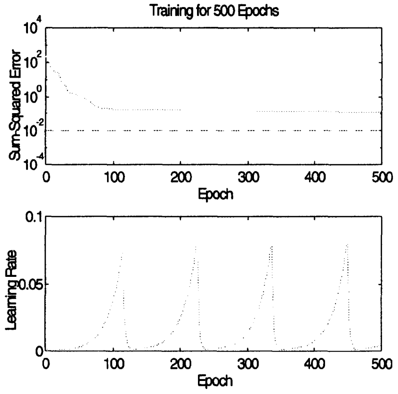


Fig. 5. The SSE and the learning rate of BP Learning Algorithm for a static ANN.

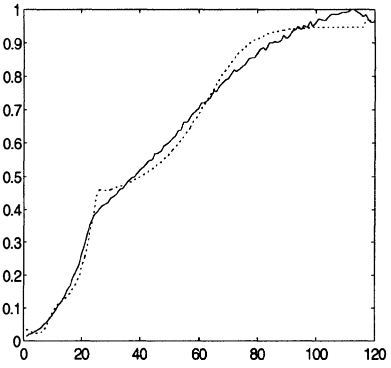


Fig. 6. Comparison of the ANN estimation of Biomass(...) Obtained from unseen data with actual biomass (solid line).

Results are presented in Figures 5 to 8. Figure 5 shows the sum of the squared errors and the learning rate during the training for modified backpropagation learning algorithm (first order optimisation method). In Figure 6 the network estimation of biomass from unseen data is compared with real data. Figures 7 and 8 show similar results for the same network trained by LM second order, nonlinear optimisation method.

Finally, a dynamical ANN is considered. As shown in Figure 9, for simplicity and ease of training, dynamical elements are located as a pre-processing layer (i.e., λ=0). The network was then trained using a second order, nonlinear optimisation method. The sum of squared error of this network (Figure 10) shows the lowest value and the fastest convergence rate; in less than 6 epochs, the sum of squared error decreased to less than 10^{-2}. Also, the network produced the best estimation of biomass from the test data (unseen data), as shown in Figure 11.

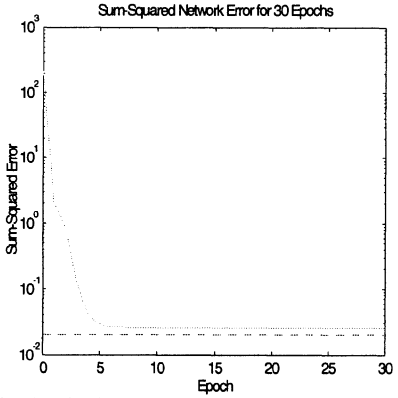


Fig. 7. The SSE of LM Learning Algorithm for a static ANN. of biomass(...)

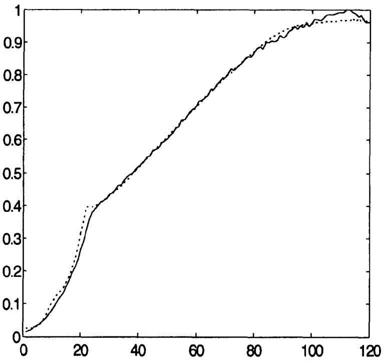


Fig. 8. Comparison of the ANN estimation Obtained from unseen data with actual biomass (solid line).

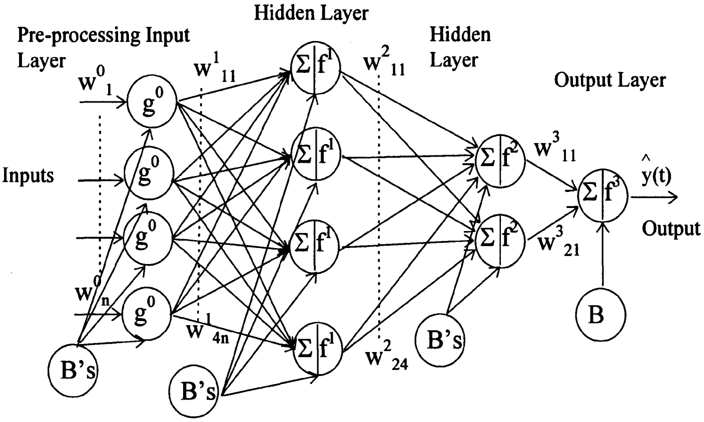


Fig. 9. A Feed-Forward ANN with Dynamical elements in pre-processing layer.

5. Conclusion

Here, as far as the authors are aware, for the fist time an artificial neural network containing dynamical elements were used as a software sensor for state-estimation and prediction of biomass concentration in a fermentation process. Practical difficulties made it impossible to use RANN. In the ANN, dynamics are not inherently included within their architectures and they are only able to produce an algebraic mapping.

Results of the different identification methods based on ANN are presented and compared. It was found that that an ANN which includes a pre-processing dynamical layer produces better results with respect to other counterparts and shows significant improvement in the convergence rate. Our experience indicate that the proposed architecture can work well in a noisy environment, partly because the dynamical elements in the pre-processing layer act like a filter.

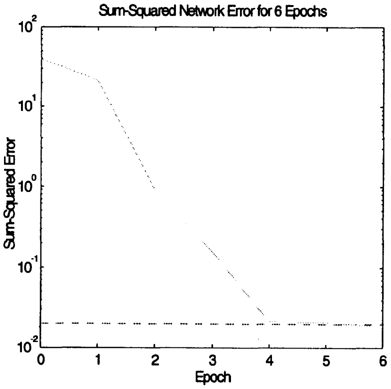


Fig. 10. The SSE of LM Learning Algorithm for a dynamical ANN.

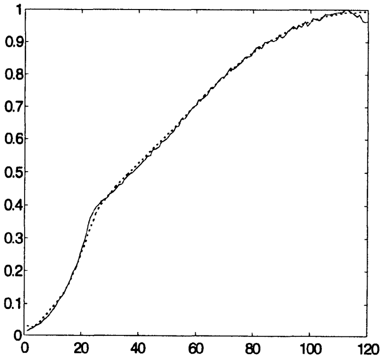


Fig. 11. Comparison of the ANN estimation of biomass(...) Obtained from unseen data with actual biomass (solid line).

State-space formulation is used to study the behaviour of dynamical ANN and classify them according to the location of non-linear element(s) in the system. It was found that ANN with a linear dynamical elements represent systems in which nonlinearities can be separated in input or output of the process. This fact can be used in design of network topology.

Further work will consist of using different architectures for dynamical ANN, comparison with other biomass estimation techniques and mathematical treatment of dynamical ANN.

6. References

1. Madigan, M.T., J.M.Martinko & J.Parker, "Biology of Micro-organisms", Prentice Hall, London,1997.
2. Fell, D.,"Understanding the Control of Metabolism", Portland Press, London,1997.
3. Soufian,Ma. and Mu. Soufian (1997) Parallel Genetic Algorithms for Optimised Fuzzy Modelling with application to a Fermentation Process, 'IEE/IEEE *Second International Conference on Genetic Algorithms in Engineering Systems: Innovations and Applications'* in September 1997, Glasgow, U.K.
4. Montague,G.A., Morris,A.J., and Tham.M.T: Enhancing bioprocess operability with generic Software Sensors, J.Biotechnology,25, 183-201, 1992.
5. Bailey,J.E., & D.F.Ollis,"Biochemical Engineering Fundamentals", MacGraw-Hill,Singapor, 1986.
6. Warnes,M.R., Glassey,J., Montague,G.A., and Kara,B: On data-based modelling techniques for fermentation processes, Process Biochemistry, 31, 2, 147-155, 1996
7. Hornik, K., Stinchcombe, M. And H. White: Multilayer Feed forward Networks are Universal Approximation. *Neural Networks,* **2**, No.5, 359-366, 1989.
8. Soufian,Ma., Mu. Soufian and M. Thomson (1997) Practical Comparison of Neural Networks and Conventional Identification Methodologies *'IEE Fifth International Conference on Artificial Neural Networks'* in July 1997, Cambridge, U.K.
9. Soufian,Ma. and Mu. Soufian (1997) On Data-based Modelling Techniques for Reciprocal Space of NMR imaging. *'The Third IFAC Symposium on Modelling and Control of Biomedical System'*, march 1997, Warwick, U.K.
10. Leigh, J.R: Fed-batch fermentation process modelling and control competition, Industrial Control Centre, University of Westminster, London , U.K. 1996.
11. Hecht-Nielsen, R. " Neurocomputing " Addison -Wesley , 1990.
12. Williams, R.J., & D. Zipser, (1989) A learning Algorithms for Continually Running Fully Recurrent Neural Networks. Neural Computation, pp 1270-1289.
13. Klopf, A.H. (1988) A neural model of classical conditioning. Psychobiology, 16 (2) pp 85-125.
14. Klopf, A.H. & J.S.Morgan, (1990) The role of time in natural intelligence: Implication of classical and instrumental conditioning for neuronal and neural network modelling. In "Learning and computational Neuroscience: Foundation of adaptive networks" Edited by Gabriel,M., & J.Moore. MIT Press, London 1990.
15. Sutton, R.S., & A.G. Barto (1990) Time-derivitives models of pavlovian reinforcements. In "Learning and computational Neuroscience: Foundation of adaptive networks" Edited by Gabriel,M., & J.Moore. MIT Press, London 1990.
16. Holden,A.V. "A Models of the stochastic activity of Neurons", Springer-Verlag, 1976.
17. Fletcher,F:*Practical Method of Optimization,* John Wiley, 1980.
18. Gill,P.E., Murray, W. and H.M. Wright: *Practical Optimization,* Academic Press, 1981.

State Estimation for Nonlinear Systems Using Restricted Genetic Optimization

Santiago Garrido, Luis Moreno, and Carlos Balaguer

Universidad Carlos III de Madrid, Leganés 28911, Madrid (Spain)

Abstract. In this paper we describe a new nonlinear estimator for filtering systems with nonlinear process and observation models, based on the optimization with RGO (Restricted Genetic Optimization). Simulation results are used to compare the performance of this method with EKF (Extended Kalman Filter), IEKF (Iterated Extended Kalman Filter), SNF (Second-order Nonlinear Filter), SIF (Single-stage Iterated Filter) and MSF (Monte-Carlo Simulation Filter) in the presence of diferents levels of noise.

1 Introduction

Whenever the state of a system have to be estimated from noisy sensor information, some kind of state estimator is employed to fuse together the data from different sensors to produce an accurate estimate of the true system state. If the system dynamics and the observation model are linear, then the minimum mean square estimate can be computed using the *Kalman Filter*. In the non-linear case the filtering problem in the state space is given by

$$x(k) = g(k, x(k-1), \varepsilon(k)), \tag{1}$$

$$z(k) = h(k, x(k), \eta(k)), \tag{2}$$

where $\varepsilon(k)$ and $\eta(k)$ are the disturbances, that are supposed to be independently distributed random vectors:

$$\begin{pmatrix} \varepsilon(k) \\ \eta(k) \end{pmatrix} \sim \left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} Q(k) & 0 \\ 0 & R(k) \end{pmatrix} \right) . \tag{3}$$

The optimal estimation implies the description of the conditional probability density

$$\hat{x}^{MMSE}(k|k) = E(x(k)|Z^k) = \int x(k)p(x(k)|Z^k)dx \ . \tag{4}$$

Unfortunately, this description needs an infinite number of parameters. For this reason, a number of suboptimal approachs have been proposed that use analitic aproximations of the probability distributions, the transition states equation or the measure equation. There are other methods, such as Monte-Carlo method, which need thousands of points in order to aproximate the conditional probability density. In aplications with many dimensions, these methods are not practical. For this reason methods with a reasonable number of operations are needed, such as the filters estudied in this work.

2 The Filtering Problem

The problem is to find an estimation of the state x(k) of the system of interest, with a non-linear discrete dynamics given by

$$x(k) = g(k, x(k-1), \varepsilon(k)) \ . \tag{5}$$

where g is the state function model, x(k) is the state of the system at time k and $\varepsilon(k)$ is the process noise.

The only information available of the system is the noisy observation given by the nonlinear measure equation

$$z(k) = h(k, x(k), \eta(k)) \ . \tag{6}$$

where $z(k)$ is the observation vector, h is the observation model that transforms the state space into the observation space and $\eta(k)$ is the measurement noise. The MMSE estimate coincides with the conditional mean. Let $\hat{x}(i|j) = E[x(i)|Z^j]$ with $Z^j = \{z(1), z(2), ..., z(j)\}^T$. The estimated covariance is $P_{xx}(i|j) = E[\{x(i) - \hat{x}(i|j)\}\{x(i) - \hat{x}(i|j)\}^T|Z^j]$. These equations are difficult to evaluate in practice. For this reason the recursive estimators are employed . In the linear case with additive gaussian noise the MMSE is the Kalman Filter. The linear updated equations in this case are

$$\hat{x}(k|k) = \hat{x}(k|k-1) + W(k)\tilde{z}(k), \tag{7}$$

$$P_{xx}(k|k) = P_{xx}(k|k-1) - W(k)P_{\nu\nu}(k|k-1)W^T(k) \ . \tag{8}$$

The vector $\tilde{z}(k)$ is called the innovation, which is the difference between the observation and the prediction:

$$\tilde{z}(k) = z(k) - \hat{z}(k|k-1) \ . \tag{9}$$

The covariance of this quantity is

$$P_{\nu\nu}(k|k-1) = P_{zz}(k|k-1) + R(k), \tag{10}$$

and $W(k)$ is the Kalman gain and its value is

$$W(k) = P_{xz}(k|k-1)P_{zz}^{-1}(k|k-1) \ . \tag{11}$$

3 Traditional Nonlinear Filters

3.1 EKF and SNF

The Extended Kalman filter is similar to a linearized Kalman Filter , with the exception that the linearization is performed on the estimated trajectory instead of a previously calculated nominal trajectory. For that reason, the functions $g(k, x(k-1), \varepsilon(k))$ and $h(k, x(k), \eta(k))$ are expanded in Taylor series around

$\hat{x}(k|k)$ with terms up to first or second order to yield the EKF or SNF respectively. The expansion with second order terms of the transition equation is:

$$\begin{aligned}
&x(k) = g(k,x(k-1),\varepsilon(k)) \simeq g(k,\hat{x}(k|k-1),0)\\
&+g_x(k,x(k-1),\varepsilon(k))(x(k-1)-\hat{x}(k-1|k-1)) + g_\varepsilon(k,x(k-1),\varepsilon(k))\varepsilon(k)\\
&+\tfrac{1}{2}\sum_{j=1}^{n} e_j(x(k-1)-\hat{x}(k-1|k-1))' g^j_{xx}(k,x(k-1),\varepsilon(k))(x(k-1)-\hat{x}(k-1|k-1))\\
&+\tfrac{1}{2}\sum_{j=1}^{n} e_j\varepsilon'(k) g^j_{\varepsilon\varepsilon}(k,x(k-1),\varepsilon(k))\varepsilon(k)\\
&+\sum_{j=1}^{n} e_j(x(k-1)-\hat{x}(k-1|k-1))' g^j_{x\varepsilon}(k,x(k-1),\varepsilon(k))\varepsilon(k)\ .
\end{aligned} \tag{12}$$

and the expansion with second order terms of the measure equation is:

$$\begin{aligned}
&z(k) = h(k,x(k),\eta(k) \simeq h(k,\hat{x}(k|k-1),0)\\
&+h_x(k,x(k-1),\eta(k))(x(k-1)-\hat{x}(k-1|k-1)) + h_\eta(k,x(k-1),\eta(k))\eta(k)\\
&+\tfrac{1}{2}\sum_{j=1}^{n} e_j(x(k-1)-\hat{x}(k-1|k-1))' h^j_{xx}(k,x(k-1),\eta(k))(x(k-1)-\hat{x}(k-1|k-1))\\
&+\tfrac{1}{2}\sum_{j=1}^{n} e_j\eta'(k) h^j_{\eta\eta}(k,x(k-1),\eta(k))\eta(k)\\
&+\sum_{j=1}^{n} e_j(x(k-1)-\hat{x}(k-1|k-1))' h^j_{x\eta}(k,x(k-1),\eta(k))\eta(k)\ .
\end{aligned} \tag{13}$$

where e_j is the j^{th} Cartesian basis vector.

3.2 IEKF

The measurement prediction, up to first order, is $\hat{z}(k|k-1) = h(k,\hat{x}(k|k-1)$. There are errors prediction errors in using $\hat{x}(k|k-1)$ for $x(k)$. Other additional errors are due to measurement nonlinearity. Its possible to alleviate these errors if the updated state is not computed as an approximate conditional mean, but a maximum a posteriori estimate.

The conditional probability density function, PDF, of $x(k)$ given Z^k, if all the random variables are Gaussian, is

$$\begin{aligned}
p(x(k)|Z^k) &= p(x(k)|z(k),Z^{k-1})\\
&= \tfrac{1}{c}p(z(k)|x(k))p(x(k|Z^{k-1}))\\
&= \tfrac{1}{c}\mathcal{N}(z(k);h(k,x(k)),R(k))\ \mathcal{N}(x(k);\hat{x}(k|k-1),P(k|k-1))\ .
\end{aligned} \tag{14}$$

Maximizig this function is equivalent to minimizing the following

$$\begin{aligned}
V(x(k)) &= \tfrac{1}{2}(z(k)-h(k,x(k)))'R(k)^{-1}(z(k)-h(k,x(k)))\\
&+\tfrac{1}{2}(x(k)-\hat{x}(k|k-1))'(P(k|k-1)^{-1}(x(k)-\hat{x}(k|k-1))\ .
\end{aligned} \tag{15}$$

The Iterated Extended Kalman Filter method (IEKF) uses a Newton-Raphson algorithm to estimate $\hat{x}(k|k)$. Expanding V in a Taylor series up to second order about the i-th estimate of x(k) results in:

$$V = V^i + V_x^{i'}(x - x^i) + \frac{1}{2}(x - x^i)'V_{xx}^i(x - x^i) \ . \tag{16}$$

Setting the gradient to zero:

$$x^{i+1} = x^i - (V_{xx}^i)^{-1}V_x^i \ . \tag{17}$$

This yields the IEKF equation:

$$\begin{aligned} \hat{x}^i(k|k) = \hat{x}^{i-1}(k|k) + P^{i-1}(k|k)H^{i-1}(k)'R(k)^{-1}\{z(k) - h(k, \hat{x}^{i-1}(k|k))\} \\ -P^{i-1}(k|k)P(k|k-1)^{-1}(\hat{x}^i(k|k) - \hat{x}(k|k-1)), \end{aligned} \tag{18}$$

with $H^{i-1}(k) = h_x(k, \hat{x}^i(k|k))$.

3.3 SIF

This technique is concerned with maximizing the *a posteriori* probability density. In the derivation of the EKF, the nominal trajectory is linearized about a nominal trajectory determined by $\hat{x}(k-1|k-1)$and a nominal state $\hat{x}(k|k-1)$. If the nominal trajectory and state are not close to the true trajectory, the truncated expansions of the EKF would represent poor approximations.

$$\begin{gathered} \hat{x}^i(k|k-1) = g^i(\hat{x}(k|k) + g_x^i(k, x(k|k-1, \varepsilon(k))))(\hat{x}(k-1|k-1) - \hat{x}(k|k-1), \\ P_{xx}^i(k|k-1) = g_x^i(k, x(k|k, \varepsilon(k))))P_{xx}^i(k|k-1)g_x^i(k, x(k|k-1, \varepsilon(k))))' + Q(k), \\ \hat{z}^i(k|k-1) = h^i(k, x(k|k-1), \eta(k)), \\ P_{zz}^i(k) = Hu^i(k, \hat{x}(k|k), x)P_{xx}^i(k, k-1)(Hu^i(k, x(k|k), x))' + R, \\ \tilde{z}^i(k) = z(k) - \hat{z}^i(k|k-1), \\ W^i(k) = P_{xx}^i(k, k-1)h_x^i(k, \hat{x}(k|k), x)'P_{zz}^i(k), \\ P_{xx}^i(k, k) = P_{xx}^i(k, k-1) - W^i(k)Pzz^i(k)W^i(k)', \\ x^i(k|k) = x^i(k, k-1) + W^i(k)\tilde{z}(k) \ . \end{gathered} \tag{19}$$

3.4 MSF

Tanizaki and Mariano proposed an algorithm based on the Monte-Carlo stochastic simulations, where the normal random numbers are generated for the error terms $\varepsilon(k)$ and $\eta(k)$ and the state variables $x(k)$ and $x(k-1)$ more precisely. This approach is based on the equations:

$$\begin{aligned} x(k) = g(k, \hat{x}(k|k-1), 0) \\ +g_x(k, x(k-1), \varepsilon(k))(x(k-1) - \hat{x}(k-1|k-1)), \end{aligned} \tag{20}$$

and

$$\begin{aligned} z(k) = h(k, \hat{x}(k|k-1), 0) \\ +h_x(k, x(k-1), \eta(k))(x(k-1) - \hat{x}(k-1|k-1)) \ . \end{aligned} \tag{21}$$

This algorithm is a combination of EKF and Monte-Carlo stochastic simulations. The equations of the Kalman filter are approximated by random draws:

$$\begin{gathered}\hat{x}(k|k-1) = \tfrac{1}{n}\sum_{i=1}^{n}\hat{x}_i(k|k-1),\\ P_{xx}(k|k-1) = \tfrac{1}{n}\sum_{i=1}^{n}(\hat{x}_i(k|k-1)-\hat{x}(k|k-1))(\hat{x}_i(k|k-1)-\hat{x}(k|k-1))',\\ \hat{z}(k|k-1) = \tfrac{1}{n}\sum_{i=1}^{n}\hat{z}_i(k|k-1),\\ P_{zz}(k|k-1) = \tfrac{1}{n}\sum_{i=1}^{n}(\hat{z}_i(k|k-1)-\hat{z}(k|k-1))(\hat{z}_i(k|k-1)-\hat{z}(k|k-1))',\\ P_{xz}(k|k-1) = \tfrac{1}{n}\sum_{i=1}^{n}(\hat{z}_i(k|k-1)-\hat{z}(k|k-1))(\hat{x}_i(k|k-1)-\hat{x}(k|k-1))',\\ W(k) = P_{xz}(k|k-1)'P_{zz}(k|k-1),\\ P_{xx}(k|k) = P_{xx}(k|k-1) - W(k)P_{zz}(k|k-1)W(k)',\end{gathered} \tag{22}$$

and

$$\hat{x}(k|k) = \hat{x}(k|k-1) + W(k)(\hat{z}_i(k|k-1) - \hat{z}(k|k-1)), \tag{23}$$

where $\hat{x}^i(k|k-1) = g(k,\hat{x}^i(k-1,k-1),\varepsilon^i(k))$, and $\hat{z}^i(k,k-1) = h(k,\hat{x}^i(k,k-1),\eta^i(k))$.

4 The New Filter

The genetic algorithm is a probabilistic process of search based on the natural selection and the genetic laws. The population $\mathcal{J} = (\mathcal{J}_1,\mathcal{J}_2,...,\mathcal{J}_n) \in J^N$ is modified according to the natural evolutionary process: after initialization, selection $\omega : J^N \longrightarrow J^N$, cross $\chi : J^N \longrightarrow J^N$, and mutation $\Xi : J^N \longrightarrow J^N$ are executed recursively in a loop. Each run of the loop is called a ***generation*** and $\mathcal{J}$ denotes the population at generation τ.

The selection operator is intended to improve the average quality of the population by giving individuals of higher quality a higher probability to be copied into the next generation. The selection therefore focuses the search on promising regions in the search space. The quality of an individual is measured by a fitness function $f : J \longrightarrow \mathbb{R}$, where J represents the space of all possible individuals.

The genetic algorithms are often used as a global optimization method of time independent functions, and usually executed off line. However, the natural selection is a local or semi local process in which the species adapt themselves to the environment, which is in turn time dependent (it is on line).

It is possible to adapt the method of genetic algorithms if the search is restricted to a neighborhood of the previous estimation using as fitness function: $f : B(\hat{(}k|k-1),\sigma) \longrightarrow \mathbb{R}$, with $\sigma = ||P(k|k-1)||$,

$$f(J) = \frac{1}{10^{-8} + V(\hat{x}(k|k-1))} \tag{24}$$

and

$$\begin{aligned}V(x(k)) = {} & \tfrac{1}{2}(z(k) - h(k,x(k)))'R(k)^{-1}(z(k) - h(k,x(k)))\\ & +\tfrac{1}{2}(x(k) - \hat{x}(k|k-1))'(P(k|k-1)^{-1}(x(k) - \hat{x}(k|k-1))\ .\end{aligned} \tag{25}$$

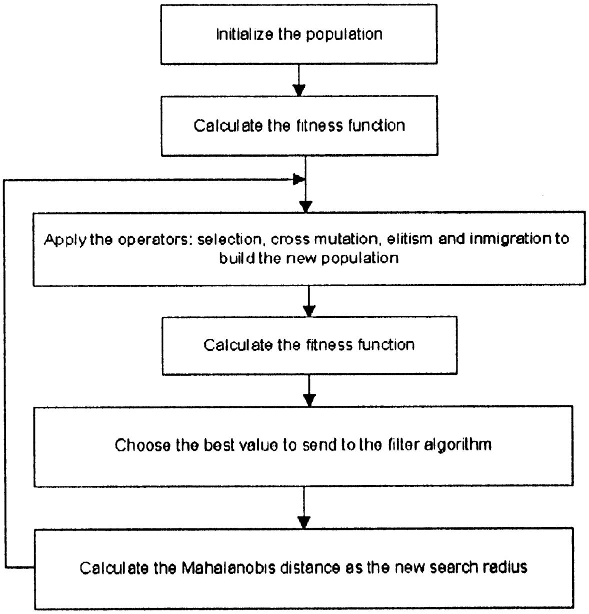

Initialize the population
Calculate the fitness function
Apply the operators: selection, cross mutation, elitism and inmigration to build the new population
Calculate the fitness function
Choose the best value to send to the filter algorithm
Calculate the Mahalanobis distance as the new search radius

Fig. 1. Restricted Genetic Optimization flowchart1.

Therefore, the algorithm updates the estimations of the states and the covariance matrix and the loop is repeated again, as illustrated in figure 1. The conditional PDF of $x(k)$ given Z^k is given as

$$\begin{aligned} &p(x(k)|Z^k) = p(x(k)|z(k), Z^{k-1}) = \\ &\tfrac{1}{c}p(z(k)|x(k))p(x(k|Z^{k-1})) = \\ &\tfrac{1}{c}N(z(k); h(k, x(k)), R(k))N(x(k); \hat{x}(k|k-1), P(k|k-1)) \ . \end{aligned} \tag{26}$$

Maximizing the above function is equivalent to calculating a maximum a posteriori (MAP) estimate. This is also equivalent to minimizing $V(x(k))$, i.e. maximizing the fitness function $f(J)$.

The standard fitness function (i.e. divided by the sum of fitness) is an approximation of the conditional density function

$$p(x(k)|Z^k) = \frac{p(z(k)|x(k))p(x(k)|Z^{k-1})}{\int p(z(k)|x(k))p(x(k)|Z^{k-1})} \ . \tag{27}$$

From the above it is clear that it is possible to calculate accurately the non linearities of the functions f and g, however introducing the hypothesis of Gaussian noise can not be avoided.

To determine the radius of the zone of search we use the Mahalanobis distance

$$d = (\hat{x}(k|k-1) - \hat{x}(k-1|k-1))'P^{-1}(k|k)(\hat{x}(k|k-1) - \hat{x}(k-1|k-1)) \tag{28}$$

that measures the uncertainty of the estimate $\hat{x}(k)$.

All this process can be applied to the Extended Kalman Filter (Restricted Genetic Optimization Filter, RGOF) or a Second-order Nonlinear Filter (Second-order Restricted Genetic Optimization Filter, SRGOF).

5 Comparison of the Nonlinear Filters

A comparison of the nonlinear filters is examined by Monte-Carlo simulations. One set of data $x(k)$ and $z(k)$ are artificially simulated and the estimate $\hat{x}(k)$ is compared with $x(k)$. and the BIAS and the RMSE between the estimated $\hat{x}(k)$ and the simulated one $x(k)$ are computed for each time k. This procedure is performed 6000 times (30 runs of 200 points each).

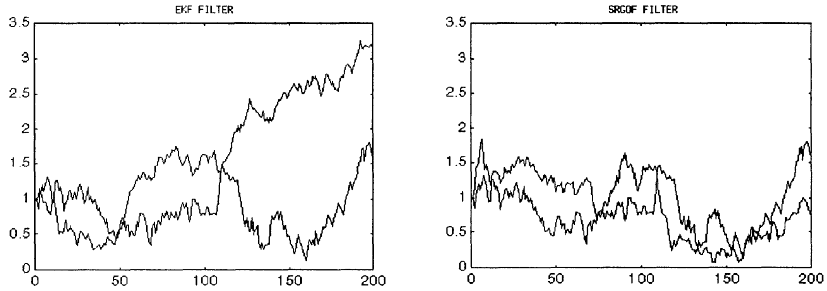


Fig. 2. Typical executions for the model 5 with EKF and SRGOF for level of noise $||\varepsilon|| = ||\eta|| = 1.0$

To compare the various estimation methods we consider the five well-known models in ascending order of nonlinearity and with three different levels of noise. In all cases $\varepsilon(k)$ and $\eta(k)$ are assumed to be normally distributed as

$$\begin{pmatrix} \varepsilon(k) \\ \eta(k) \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} C & 0 \\ 0 & C \end{pmatrix}\right), \tag{29}$$

where C is a constant and the initial value $x(0)$ is distributed as a normal random variable.

The first model is linear:

$$\begin{cases} x(k) = x(k-1) + \varepsilon(k) \\ z(k) = x(k) + \eta(k) \ . \end{cases} \tag{30}$$

The second one is the Logistic Model:

$$\begin{cases} x(k) = \frac{exp(x(k-1))}{exp(x(k-1))+exp(\varepsilon(k))} \\ z(k) = \frac{exp(x(k))}{exp(x(k))+exp(\eta(k))} \ . \end{cases} \tag{31}$$

The third one is the ARCH model:

$$\begin{cases} x(k) = (1 - b + b(x(k-1))^2)^{1/2} + \eta(k) \\ z(k) = x(k) + \varepsilon(k) \ . \end{cases} \tag{32}$$

The fourth one is the Nonstationary Growth Model given by Kitagawa (1987) and Carlin et al. (1992).

$$\begin{cases} x(k) = 0.5x(k-1) + \frac{25x(k-1)}{1+x^2(k-1)} + 8cos(1.2(t-1)) + \varepsilon(k) \\ z(k) = \frac{x^2(k)}{2}0 + \eta(k) \ . \end{cases} \tag{33}$$

The last one correspond to the Tracking, with an angle-only sensor given by Bar-Shalom and Fortmann (1988)

$$\begin{cases} x(k) = \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix} x(k-1) + \varepsilon(k) \\ z(k) = atan\left(\frac{20}{(x(1,k) - x_p(k))}\right) + \eta(k) \ . \end{cases} \tag{34}$$

Table 1. Comparison of BIAS and RMSE for level of noise $||\varepsilon|| = ||\eta|| = 0.1$. Each number is the mean of thirty runs of two hundred points each.

	Model 1		Model 2		Model 3		Model 4		Model 5	
	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE
EKF	-0.2062	0.9616	0.0058	0.0876	-0.1275	0.9162	0.7118	10.9863	-0.2342	1.1963
IEKF	-0.7654	2.0934	0.0250	0.0822	-1.2593	2.1868	0.4796	10.8942	-0.2488	1.0293
SNF	-0.2063	0.9638	0.0109	0.0536	-0.1210	0.8812	0.4796	10.8942	-0.4361	0.9003
SIF	0.1955	1.9693	8.5E-05	0.0690	-0.1812	1.0947	0.5861	13.1817	-0.3238	0.8980
MSF	-4.9743	36.5653	0.0057	0.0786	-0.1397	0.8718	0.4487	11.3058	-0.2453	0.8828
RGOF	-0.1889	0.9515	0.0076	0.0532	-0.1142	0.8695	0.2688	10.3893	0.2411	1.1523
SRGOF	0.6469	1.3145	0.0078	0.0543	-0.1198	0.8554	0.3893	10.1435	0.0938	0.8511

6 Discussion

In this paper we have described a new method for filtering nonlinear systems. This method uses Restricted Genetic Optimization to reduce the estimation error of the prior EKF or SNF estimation.

Tables 1-3 summarize the results of the simulations of the five models with three levels of noise: $||\varepsilon|| = ||\eta|| = 0.1$, 0.5 and 1.0. Each number shown in the tables represents the mean of thirty runs of two hundred points each.

It is clear from the above results that the precision (based on the RMS Error criteria) of the filter estimates of the SRGOF are certainly improved over the other algorithms, especially in the noisier and more nonlinear situations. Judging from the criteria of BIAS there are no big differences between the different

Table 2. Comparison of BIAS and RMSE for level of noise $||\varepsilon|| = ||\eta|| = 0.5$. Each number is the mean of thirty runs of two hundred points each.

	Model 1		Model 2		Model 3		Model 4		Model 5	
	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE
EKF	-1.9802	4.996	0.2251	1.5537	-0.1691	0.8632	0.2727	14.7955	0.1648	5.5196
IEKF	-6.7135	10.068	0.2233	1.5561	-0.7962	1.7991	0.1858	14.7742	-0.1300	5.0794
SNF	-0.7908	4.354	0.2275	1.5546	-0.1250	0.8463	0.1858	14.7742	-0.0797	4.6296
SIF	0.4687	8.767	0.2269	1.5494	-0.1911	1.0864	0.8650	17.9635	0.1989	4.4094
MSF	-59.0465	154.528	0.2236	1.5540	-0.0628	0.8523	0.5375	14.8041	-0.0202	5.1212
RGOF	-0.8040	3.847	0.2209	1.5525	-0.1453	0.8573	0.4761	10.3697	0.3438	5.9636
SRGOF	-1.1171	4.2704	0.2055	1.5063	-0.1603	0.8611	-1.2549	6.0052	-1.0691	4.8815

Table 3. Comparison of BIAS and RMSE for level of noise $||\varepsilon|| = ||\eta|| = 1.0$. Each number is the mean of thirty runs of two hundred points each.

	Model 1		Model 2		Model 3		Model 4		Model 5	
	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE	BIAS	RMSE
EKF	-1.0107	4.8071	0.0811	0.2457	-0.1275	0.9162	0.28454	14.5453	0.2183	11.0601
IEKF	-4.0117	9.7702	0.1628	0.2791	-1.2593	2.1868	0.19430	14.5721	0.2765	9.7927
SNF	-1.0325	4.8194	0.0859	0.2374	-0.1210	0.8812	0.19430	14.5721	0.0084	8.2852
SIF	0.9615	9.8352	0.0369	0.3276	-0.1812	1.0947	0.52118	15.6697	-1.6102	8.2340
MSF	-17.5458	173.3967	0.0810	0.2417	-0.1397	0.8718	0.43729	14.6868	0.6963	8.1326
RGOF	-2.3808	5.5300	0.0524	0.2286	-0.1146	0.8696	0.01695	10.1305	0.3160	10.2766
SRGOF	-2.1398	5.9861	0.0529	0.2287	-0.1198	0.8554	0.11951	9.9886	1.7845	9.1265

methods, however some of the algorithms (MSF, SIF and IEKF) are less robust in certain situations.

Finally, the most important charactheristic of the proposed SRGOF algorithm is its robustness in the noisier and highly nonlinear situations which are the desired characteristics of such filters.

References

1. Armingol, J.M., Moreno, L.E., Garrido, S., de la Escalera, A., Salichs, M.A.: Mobile Robot Localization Using a Non-Linear Evolutionary Filter. 3rd IFAC Symp. on Intelligent Autonomous Vehicles (IAV'98)(to appear).
2. Bar-Shalom, Y., Fortmann, T.E.:Tracking and Data Association. The Academic Press, (1988).
3. Bar-Shalom, Y., Xiao-Rong, L..:Estimation and Tracking. Artech House, (1993).
4. Julier, S.J., Uhlmann, J.K., Durrant-Whyte, H.F.: A New Approach for Filtering Nonlinear Systems. A new approachfor the nonlinear transformation of means and covariances in linear filters. IEEE Trans. on Automatic Control (1996).

5. Kitagawa, G.:Non-Gaussian State-Space Modeling of Nonstationary Time Series. Journal of the American Statistical Association, **82** (1987) 1032-1063.
6. Maybeck, P.S.: Stochastic models, Estimation, and Control, volume 2. Academic Press (1982).
7. Tanizaki, H., Mariano, R.S.:Nonlinear Filters Based on Taylor Series Expansions. Communications in Statistics, Theory and Methods, **25** (1996) 6.
8. Wishner,R.P., Tabaczynski,J.A., Athans,M.:A Comparison of Three Non-Linear Filters. Automatica, **5** (1969) 487-496.

Identification of a Nonlinear Industrial Process Via Fuzzy Clustering

B.Moshiri[1,2] ,and, S.Chaychi. Maleki [1]
[1] Dept of Electriacl and computer Engineering,University of Tehran, Tehran, Iran.
[2] Intelligent Systems Reasearch Faculty, IPM,Tehran, Iran.
e-mail;Moshiri@karun.ipm.ac.ir

Keywords: Fuzzy, Clustering, Nonlinear Process, Modelling.

Abstract. This paper presents a fuzzy logic approach to complex system modeling that is based on fuzzy clustering technique. As compared with other modeling methods,the proposed approach has the advantage of simplicity, flexibility, and high accuracy. Further, it is easy to use and may be handled by an automatic procedure. An industrial process example (i.e; Heat exchanger) is provided to illustrate the performance of the proposed apprach.

1. Introduction.

THERE are many reasons why a model of process may be required. It could, for example, be used to simulate the real process or it could be used to design a controller. However, it is not, in general, an easy task to find a global universal structure, as has been done in the conventional modeling techniques, for a complex process. Even if such a structure is available, the order of a model is often unrealistically high, due to non linearity of the process.

In recent years fuzzy logic based modeling , as complement to conventional modeling techniques, has become an active research topic and found successful application in many areas. However, most present models/controllers have been built based only on operator's experience and knowledge.When a process is complex, there may not be any expert who can control the process well. In this kind of situation, it is not possible to use operator's experience and knowledge. Instead, it is desirable to build a mathematical model of the process which enables us to derive the control rules theoretically. Further, if there is no reason to believe that the operator's control is optimal, it may be preferable to develop a model based control strategy as in conventional control theory.

Among different fuzzy modeling approaches, the fuzzy model suggested by Takagi and Sugeno is excellent in system description and useful for model based control [1]. However it is difficult to implement and time consuming due to complexity of algorithm. That is, when the number of input variables is large, the number of the possible structure becomes combinationally large.

In our approach, we have identified dynamic model of the nonlinear process using only input-output data.This stage of modeling is usually referred to as identification. Then, fuzzy clustering approach is investigated, and the following sections will give the main ideas of proposed technique.

2. Fuzzy model.

As the expression of a fuzzy model,we use the following equations:

$$L^i : If\ x_i \ \ is\ A_1{}^i\ and\ x_2\ is\ A_2{}^i,...,x_m\ \ is\ A_m{}^i$$
$$Then\ y = y^i \qquad (2\text{-}1)$$

$$y^{k=} \frac{\sum_{i=1}^{n} w^i y^i}{\sum_{i=1}^{n} w^i} \qquad where\ \ w^i = \prod_{j=1}^{m} A_j{}^i\,(x_j{}^k) \qquad (2\text{-}2)$$

where L^i (i=1,2,...,n) represents i-th implication and x_j (j = 1,2,...,m) an input variable and y^i , the output from the i-th fuzzy implication, $A_j{}^i$ is membership grade ,n,m are the number of clusters and variables respectively.

The identification of a fuzzy model using input-output data is divided in two stages: *Structure* and *parameter* identification. The former consists of premise structure and the latter consists of consequent parameter identification. The consequent parameters are the outputs of the fuzzy implications. In our approach *nearest neighborhood clustering* method,have been used for generation of fuzzy clusters [2]. The advantage of this method is that it provides an automatic way of forming of the reference fuzzy sets and does not require any initial knowledge about the structure in data set. Given a finite set of objects A={ x_1 , x_2 ,. . .,x_m }, where $x_j \in$ R , the fuzzy clustering of A into n clusters is a process of assigning grades of membership of each object to every cluster. The recursive steps in the identification of a fuzzy model are as follows [3]:

a- Choice of premise structure
b- Premise parameter identification
c- Choice of consequent structure
d- Consequent parameter identification
e- Calculation of membership grades
f- Verification of consequent structure
e- Verification of premise structure

3. Problem formulation.

Suppose that we are given N input-output pairs (X^l,y^l) , where l = 1,2,...,N and X^l is input vector, $X^l = (x_1{}^l, x_2{}^l, ..., x_m{}^l)$. Our task is to construct a fuzzy logic system g (X^l) which can match all the N pairs to any given accuracy. That is , for any given ε >**0** , we require that $|g\ (X^l) - y^l\ | < \varepsilon$ for all l= 1, 2, ..., N .

The optimal fuzzy logic system is constructed by equation (2-2) with n=N.

This fuzzy logic system uses one rule for one input-output data in the training set, thus it is no longer a practical system if the number of input-output pairs in the training set is large. For these large -sample problems, various clustering techniques can be used to classify the samples so that each class can be represented by only one rule in the fuzzy logic system.Here we use the simple nearest neighborhood clustering scheme. In this case, the number of classes is unknown and classification by clustering is actually to construct the probability densities from pattern samples. *Adaptive sample set construction* is one of the approaches commonly used. The whole procedure consists of the following steps [4,2]:

1) Starting with the first input-output pair, establish a cluster center Z^1 at X^1 and set $Y^1(1) = y^1$, $B^1(1) = 1$.Select a radius r.

2) Suppose that when we consider the k th input-output pair (X^k, y^k), $k = 2, 3, \ldots,$ there are M clusters with centers at $Z^1, Z^2, \ldots, Z^M$ and number of elements in each cluster $N_c^{\,1}, N_c^{\,2}, \ldots, N_c^{\,M}$.
Compute the distance of X^k to these M cluster centers, $|X^k - Z^l|$, $l = 1, 2, \ldots, M$ and let the smallest distance be $|X^k - Z^{lk}|$, that is , the nearest cluster to X^k is Z^{lk}.
Then:

a) If $|X^k - Z^{lk}| > r$, establish X^k as new cluster center $Z^{M+1} = X^k$, set $Y^{M+1}(k) = y^k$, $B^{M+1}(k) = 1$, and keep $Y^l(k) = Y^l(k-1)$, $B^l(k) = B^l(k-1)$ for $l=1, 2, \ldots, M$.

b) If $|X^k - Z^{lk}| \le r$ do the following:

$$N_c^{\,lk} = N_c^{\,lk} + 1 \qquad (3\text{-}1)$$

Define:

$$\lambda_{lk} = (\,(Nc^{lk} - 1\,)\,/\,Nc^{lk}\,)) \qquad (3\text{-}2)$$

And set:

$$Z^{lk} = \lambda_{lk}\, Z^{lk} + (\,1 - \lambda_{lk}\,)\, X^k \qquad (3\text{-}3)$$

$$\Sigma^{lk} = \lambda_{lk}\, \Sigma^{lk} + (1-\lambda_{lk})\, diag((X^k - Z^{lk})^T (X^k - Z^{lk})) \qquad (3\text{-}4)$$

$$Y^{lk}(k) = \lambda_{lk}\, Y^{lk}(k-1) + (\,1-\lambda_{lk}\,)\, y^k \qquad (3\text{-}5)$$

$$B^{lk}(k) = \lambda_{lk}\, B^{lk}(k-1) + 1 - \lambda_{lk} \qquad (3\text{-}6)$$

And keep:

$Y^l(k) = Y^l(k-1)$, $B^l(k) = B^l(k-1)$ for $l=1, 2, \ldots, M$ with $l \neq lk$.

Where Σ^j ($j = 1, 2, \ldots, M$) is the variance vector of the j th cluster.

3) The adaptive fuzzy system at the k th step regarding Gaussian membership function (bell-type) is computed as :

$$y^k = \frac{\sum_{i=1}^{M(k)} Y^i(k)\, w^i(k)}{\sum_{i=1}^{M(k)} B^j(k)\, w^i(k)} \qquad (3\text{-}7)$$

where $$w^i(k) = \prod_{j=1}^{m} A_j^{\,i}(x_j(k)) \qquad (3\text{-}8)$$

and $$A_j^{\,i}(x_j(k)) = exp\{-((x_j(k) - z^i_j)/\sigma_j^{\,i})^2)\} \qquad (3\text{-}9)$$

where z^i_j and $\sigma_j^{\,i}$ are the elements of cluster centers and variances.

4) Repeat steps **2** and **3** until all pattern samples have been assigned . There would be some reassignment of X^k when again passed through the system. The reason is that the means and variances have been adjusted with each X^k assigned to Z^i.

5) After the training is considered to be completed (that means that X^k no longer changes the belonging class), we can let the system go freely to do the clustering on a large number of pattern samples.

We now make some remarks on this adaptive fuzzy system.

Remark 3.1. Because for each input-output pair a new cluster may be formed , this adaptive fuzzy system performs parameter adaptation as well as structure adaptation.

Remark 3.2. Because the cluster center and variance may be changed , number of clusters in this case ,is less than ,the situation of clusters with fixed centers.

Remark 3.3. There would be some reassignment of X when again passed through the system, due to changing the center and variance of clusters.

Remark 3.4. The parameter λ can be viewed as forgetting factor, in such a case that fast adaptation may be required we can replace this parameter by $\alpha\lambda$ where $0<\alpha\leq 1$.

4. Simulation Results.

i-Fluidized bed reactor

The reactor design and model is that studied by McAlluy *et al.*[5]. As shown in *Figure*.1, the feed to the reactor consists of ethylen,comonomer, hydrogen, inerts, and catalyst. A stream of unreacted gases flows from the top of the reactor and is cooled by passing through a heat exchanger in counter-current flow with cooling water. Cooling rates in the heat exchanger are adjusted by instantaneously blending cold and warm water streams while maintaining a constant total cooling water flowrate through the heat exchanger. In this paper ,the fuzzy modeling of gas loop heat exchanger is considered, to achieve good control law .

ii-Heat exchanger model

The external heat exchanger shown in *Figure.*1 is a single-pass shell and tube gas cooler. Gas flows on tube side, counter-current to cooling water on the shell side. An accurate representation of the heat removal system can be obtained using staged heat exchanger model. An N-stage heat exchanger is illustrated in *figure.*2 . The single-pass heat exchanger has been divided into a number of 'lumps' of equal heat transfer area [6]. In this example the conventional log-mean temperature difference (**LMTD**) driving force model is considered.

iii-LMTD Model

The following energy balance equations can be written for each of the N stage of a single-pass counterflow exchanger with a log-mean temperature difference driving force for heat transfer between the shell and tube sides.

$$dT_{w,j} / dt =(NF_w /M_w)(T_{w,j-1} -T_{w,j})-(UA/M_wC_{pw})\Delta T_{lm,j} \qquad (4\text{-}1)$$

$$dT_{g,j} / dt =(NF_g /M_g)(T_{g,j+1}\text{-}T_{g,j})+ (UA/M_gC_{pg})\Delta T_{lm,j} \qquad (4\text{-}2)$$

where:

$$\Delta T_{lm,j} = \frac{(T_{w,j}-T_{g,j+1})-(T_{w,j-1}-T_{g,j})}{Ln\dfrac{(T_{w,j}-T_{g,j+1})}{(T_{w,j-1}-T_{g,j})}} \qquad (4\text{-}3)$$

$\Delta T_{lm,\,j}$ is the log mean temperature difference between the water and the gas in the ***j*** th stage. M_w and M_g are the total molar holdups in the water and gas sides of the heat exchanger, respectively. $T_{g\,,N+1}$ ($=T$) is the inlet gas temperature and $T_{w\,,0}$ (=0) $T_{w\,i}$ is the inlet cooling water temperature to the external heat exchanger. $T_{g\,,j}$ and $T_{w\,,j}$ denote the temperature of the gas and coolant, respectively, leaving the ***j*** th stage of the heat exchanger. For simulation of this exchanger we have used design data of this exchanger for the same gasphase Poly-Ethylene plant. The exchanger is supposed to have two stages , inlet gas temperature is about 82 deg C and water inlet temperature is about 27 deg C, the output of this plant is outlet temperature of the gas. Design data for this exchanger are: M_w = 4057 kg, F_w = 286.111kg/sec , U = 55.72 cal/sec.m^2 .deg C . A =734 m^2 ,$C_{p\,w}$ =1000 cal/kg.deg.C ,M_g =102.75kg , F_g

=133.366 kg/sec , C_{pg} =340 cal/kg.deg C ,and sampling time is considered to be 0.1 sec.

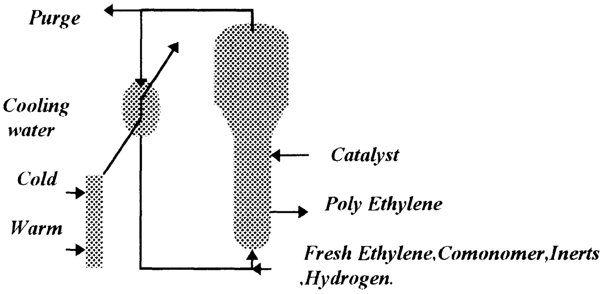


Figure 1 . ***Gasphase Poly Ethylene Reactor***

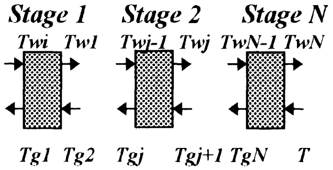


Figure 2 . ***N stage gasloop cooler.***

Using the input and output data generated by real model and the procedure suggested by this paper, the construction of fuzzy model for the given system is simulated. In this simulation the outlet temperature of gas is the system output, and input vector elements are the temperatures of gas and water in different stages and flow of water which is the control output for this system.

The discrete model of this system in general form can be viewed as $y^{k+1}=f(X^k)$, where $X^k=(x_1^k, x_2^k,..., x_m^k)$. 300 simulated data points are generated from the plant model. The first 150 data points are obtained by assuming a random input signal uniformly distributed in the interval [41,82] for the gas inlet temperature and [143,286] for the water flow and [20,27] for the water inlet temperature, and the last 150 data points are obtained by using a sinusoid input signal for flow of water. We use the first 150 data to train the system, and to construct the fuzzy model. The performance of the fuzzy model is tested using the remaining 150 data point. *Figure 3.* shows the output of identified model and the actual model, as well as the identification error. It can be seen that the output of identified model attains rather good match with that of the actual model, not only in the modeling term but also in the testing term. It can be expected that the performance of the identified fuzzy model may be further improved if the number of data points used to build the model is increased.

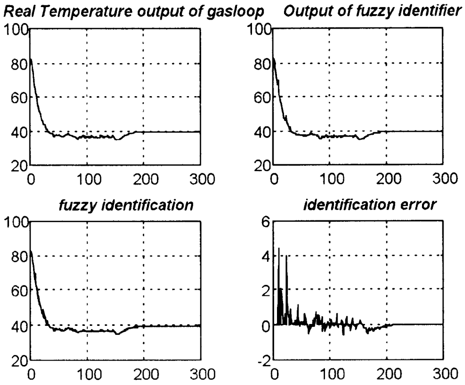


Figure 3. ***Output of the identified model and the actual model.***

5. Concluding remarks.

This paper developes a fuzzy logic based approach to complex system modeling, in which fuzzy clustering technique has played an essential role. This approach is simple in practice,but the resulting outcome is interesting. A great advantage of this approach is simultaneous parameter and structure adaptation. Comparison with fixed parameter clustering method shows that utilizing this approach yields to better accuracy with reduced clusters, as shown in following table;*Table 1.*

Criteria:→ *Approach:↓*	*Max imum identification error*	*Number of clusters*	*Forgetting factor*
Fixed parameters clustering	*14.1646*	*28*	*0.96*
This approach	*4.397*	*24*	*0.96*

Table 1. ***Comparison of two clustering method with the same initial variance.***

References.

[1] T. Takagi and M.Sugeno,'Fuzzy identification of systems and its Applications to Modeling and Control.' IEEE Transaction on Systems,Man,and Cybernetics, Vol, SMC-15,No.1,pp116-132,(1985).

[2] L.X.Wang' Training of fuzzy logic systems using nearest neighborhood clustering'. Adaptive Fuzzy Systems and Control. 1st edn,Printice-Hall,(1994).

[3] M.K.Park,S.Hwan. A new identification method for a fuzzy model. IEEE conference on Fuzzy systems. (1995)

[4] S.Bow,'Pattern recognition and image processing,'Dekker,(1992).

[5] K.BMcAuley,AICHE ,41(4),868,(1992).

[6] W.LLuyben,Process Modeling, Simulation,and Control for Chemical engineers, 2nd edn, McGraw-Hill, New York,49,(1990).

Applying Computer Vision Techniques to Traffic Monitoring Tasks

Jorge Badenas and Filiberto Pla

Dept. Informática, Universitat Jaume I, Castellón 12071 (SPAIN)
Telf: +34-964-345680 Ext. 4714
(badenas,pla)@inf.uji.es

Abstract. This paper presents a method[1] for tracking and segmenting vehicles in a traffic scene. The approach is based on a frame to frame segmentation followed by a tracking process. As opposed to usual segmentation methods, our method feedbacks segmentation with tracking information for improving results. Several facilities are provided for traffic monitoring such as vehicles trajectories surveillance, segmentation of vehicle shape, measuring the mean velocity of the traffic, counting the vehicles that are moving on the lanes of a road or a motorway, counting the vehicles that stop at a junction and detection of events such as a vehicle stops on a road or a possible accident.

Key Words : Region tracking, motion analysis, motion segmentation, traffic monitoring, vehicle surveillance

1 Introduction

In recent years, as result of advances in information techniques both in terms of computational power and cost, it has become possible to use computer vision to perform many everyday tasks. This paper shows how it can be applied to traffic monitoring substituting existing methods such as Inductive Loop Detectors and Microwave Vehicle Detectors which are more expensive, and have more limited usefulness.

A typical traffic scene consists of untextured objects with a regular shape moving over a surface which is also untextured. In this context, a traffic monitoring system has to survey moving objects detecting anomalous behaviors and situations, and producing measures.

Several works have been reported on the application of computer vision to traffic monitoring. One of the early works [1] describes a system with motion analysis based on simple frame differencing. This simple approach can not provide enough information and produces poor results due to it is very sensitive to noise. Dubuisson and Jain [2] proposed a technique for segmenting vehicles also based on image substraction, but combined with color segmentation of regions.

[1] The present work was partially supported by the projects ESPRIT EP-21007 IOTA, CICYT TIC95-0676-C02-01 and Generalitat Valenciana GV97-TI-05-26

This technique can not deal with occlusions since it does not make motion estimation. Hence, it distinguishes moving regions from stopped regions, but not regions with different motions. Badenas et al [3] presented another method based on regions. This method is a combination of several techniques which produces motion estimation and segmentation of vehicles. Thus, this approach can cope with occlusions and carry out several monitoring tasks such as counting vehicles and measuring traffic fluency.

Another group of methods fits models to moving vehicles. These methods are suitable for very structured scenes, such as roads, but require a high computational cost which reduces their usefulness in traffic monitoring. We can distinguish two main tends on these methods: methods which fit 3-D models of generic vehicles [4], [5], and methods which track 2-D contours onto the image plane [6], [7]. The method [4] fits wireframe models to vehicles which can move with three degrees of freedom. A similar approach is shown in [5], where the edges extracted from the image are matched to 3-D segments of a generic car model that is projected onto the image plane. This matching is only applied to edges moving with coherent motion, to avoid mixing up edges from different vehicles. 2-D models have a lower computational cost, but present problems when the size and the number of vehicles increase. In [7], B-splines with four control points are fitted to vehicles projection. Weber at al. [6] method describe the contours by a closed cubic spline and employs two Kalman filters, one for estimating the affine motion parameters and other one for estimating the shape of the contours.

A different group of methods for motion analysis tracks single tokens such as points or lines, that are extracted from the image,[8], [9]. These methods present problems due to they do not provide explicit grouping of tokens moving with coherent motion and are quite sensitive to noise.

We propose in this paper an approach based on tracking regions which integrates information recorded over a sequence on a frame by frame basis for improving segmentation. For every frame of the sequence, our method carries out a frame to frame motion segmentation that is followed by a tracking process. This process matches regions segmented in previous frames to regions segmented in the current frame. Matching allows the system to know the evolution of the regions. This evolution is recorded by using a mask strategy that enable us to correct frame to frame segmentation of the current frame. This correction avoids both sudden appearance and disappearance of regions or parts of regions from one frame to the next frame.

Our method provides traffic monitoring facilities such as counting vehicles that are moving on each road lane, counting vehicles that have stopped at junctions, surveillance of vehicles trajectories, estimation of mean velocity of the traffic and detection of traffic jams. In addition to these facilities that could be referred to normal facilities, anomalous or dangerous situations can also be detected such as vehicles that stop on the road, vehicles that cross to a forbidden lane, vehicles that go out the boundaries of the road or a crash of vehicles.

The rest of the paper is organized as follows: section 2 explains the parts of the algorithm that allows us to segment vehicles and survey their trajectories, section

3 describes the traffic monitoring facilities provided by the method, section 4 shows results over two traffic sequences, and section 5 presents conclusions.

2 The tracking algorithm

The purpose of the algorithm is to segment the shape of the vehicles and to track them along a sequence. Figure 1 shows the different parts of the developed algorithm. Initially, the first frame is segmented using the current frame and its consecutive one, and the segmented regions are inserted in a list called the *master list*. Then, several steps are repeated for each new frame of the sequence. First a frame to frame segmentation that generates a list of new regions is carried out, then tracking step matches each region from the master list (master region) to the most similar new region. Hence, matching establishes a correspondence between the apparitions of a same region in several frames of the sequence. When a new region is not matched, it is inserted in the master list, since it can be a new vehicle. The next step recovers lost regions that have regularly appeared in previous frames and that illogically have disappeared in the current frame to frame segmentation. Updating master list step adds for each master region information from current segmentation. For each region from the master region, a matrix is stored. This matrix is updated after each frame to frame segmentation and records for each pixel of the image how many times it has belonged to the region during recent segmentations. Hence, each of these matrixes contains the evolution of the shape of its corresponding region. At the next step, information stored for each region is used to produce the final segmentation of the current frame. The next step removes from master list, the regions that have not matched to new regions in recent iterations. The last step applies the motion estimated for each region to the matrix of each master region. Thus, the values stored in each matrix for the pixels of each region are moved to the place where they will be found in the next frame to frame segmentation.

2.1 Motion segmentation using two consecutive frames

This part carries out an initial segmentation for every frame of the sequence. The frame is segmented into regions of similar intensity by using a clustering algorithm.

The next step is region-based motion estimation. Motion estimation is performed with a robust kernel with subpixel accuracy and by a mutiresolution approach. This step involves finding for every region the translation parameters that minimises the sum of displaced frame differences, transformed by a robust kernel ρ. The summation is over all pixels from the reference region. In fact, we are minimising an error measure E defined as follows:

$$E_i(dx, dy) = \frac{1}{N_j} \sum_{(x,y)\in C_j} \rho(I_1(x,y) - I_2(x + dx, y + dy), \alpha, \lambda) \tag{1}$$

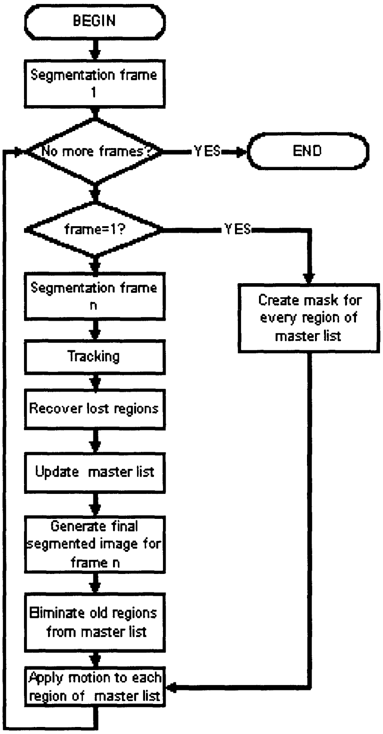

BEGIN
Segmentation frame 1
No more frames?
YES
END
frame=1?
YES
Segmentation frame n
Create mask for every region of master list
Tracking
Recover lost regions
Update master list
Generate final segmented image for frame n
Eliminate old regions from master list
Apply motion to each region of master list

Fig. 1. Algorithm of the method

where C_j is the region j, $I_1(x,y)$, $I_2(x,y)$ are the pixel intensity values at location (x,y) in the reference and consecutive frames respectively. $\rho()$ is the robust redescending function and α, λ are the function parameters.

Motion segmentation is performed after estimating motion parameters. First, adjacent regions with coherent motion are merged. Next, for each pair of adjacent regions we calculate the error for each region when the motion of the other one is applied. If one of these errors for a region is lower or similar to the error that it had with its estimated motion, then both regions are merged. For a complete description of the frame to frame segmentation we refer the reader to [3].

2.2 Tracking regions

Tracking enables the system to follow the trajectories of the segmented regions. Tracking is based on matching regions from the master list to regions included in the list of regions segmented in current frame to frame segmentation.

Each master region is matched to the most similar new region considering five features: x and y region means, intensity mean and motion parameters. These features define a vector $V = (m_x, m_y, m_z, v_x, v_y)$ for each region that is used for looking for that region that minimizes a weighted squared Euclidean distance:

$$D^{mn} = (V^m - V^n)^t W (V^m - V^n) \tag{2}$$

where V^m is the features vector of the master region R_m, V^n is the features vector of the new region R_n, and W is a 5x5 diagonal matrix that contains a weight for each feature $(w_x, w_y, w_z, w_{v_x}\ and\ w_{v_y})$.

2.3 Integrating information from several frame segmentations

The information that is recovered from each motion segmentation and related by tracking is used at two levels: pixel level and region level.

Pixel level. Every time a region R_m is inserted in the master list, a matrix M^m of image size [2] is created. Each matrix cell contains two values (v and o) that are initialized to zero. Each time that a master region R_m matches to a new region R_n, the values v of the matrix of R_m that coincide with the pixels that belong to the region R_n are incremented by one. Moreover, for these pixels, o is set to zero. The value v contains for each pixel the number of times that it has belonged to a region.

The value o contains the number of consecutive times in which a pixel has not belonged to a region. Thus, for every region R_m, each time that a pixel with a value v larger than zero in the matrix M^m does not belong to the region R_n that has been matched to R_m the value o is incremented for that pixel. When a value o of a pixel reaches a value MAX_O, its corresponding value v is also set to zero.

The maximum value that each v can reach is limited by a constant MAX_V. In order to avoid that the background takes all the pixels of the image, for this region MAX_V is lower than for the rest of regions.

After updating the matrixes of the regions the final segmentation of the frame is carried out. Each image pixel is assigned to that region with a maximum value v in that pixel.

Region level. In some frames, a region that has been appearing for several frames unlogically disappears. Our method counts the number of frames in which each master region is matched to another region, and when this number is larger than certain threshold q, the region is declared as a *vehicle region.* It means that a vehicle region can not disappear except when it goes out the image boundaries. The method uses the information stored in the matrixes for recovering its shape. This fact happens when a vehicle stops or when it is going away from the camera, its motion tends to be similar to the one of the neighbouring regions (cars or

[2] Actually only a matrix of the minimum rectangle size that encloses the region is required. For the sake of simplicity, we will consider that its size is the same that the image size.

background). In this case, a frame to frame segmentation can not separate the vehicle region from its neighbouring regions.

If a *vehicle region* R_m is not matched to another region, it is considered as *lost region.* The algorithm looks for what segmented region has taken its pixels and what master region has been matched this new region to. The region that has taken the pixels can be found out by analysing the matrix of the *lost region.* For each new segmented region R_n an accumulator A_n is created:

$$\forall I_{ij} \in Image/I_{ij} \in R_n, \quad A_n = A_n + M_{ij}^{R_m}.v \tag{3}$$

The largest A_n corresponds to the region R_n that has taken the pixels that would belong to the lost region if a good segmentation had been made. Supposing that R_n was matched to R_k, then a new region R_t that is matched to the lost region R_m is created in the segmented frame. R_t recibes the pixels (i,j) that were assigned in the frame to frame segmentation to R_n with $M_{ij}^{R_m}.v > M_{ij}^{R_k}.v$, where $M_{ij}^{R_x}.v$ is the value v in the matrix M of the region x.

When a region do not reach that value q in a few frames, it is removed, since the system considers that it appeared due to a bad segmentation.

3 Traffic monitoring facilities

For a traffic monitoring system the following basic facilities are provided: counting vehicles(traffic mean velocity), vehicle trajectories surveillance, detection of stopped vehicles and detection of anomalous events. To carry out some of these tasks it is needed to delimit the road and lane boundaries. This can be automatically achieved by applying the technique exposed in [10].

3.1 Vehicle trajectories surveillance and counting the number of vehicles

Trajectories surveillance is directly achieved from tracking. This allows us to know the position of every vehicle at each frame and to count the number of vehicles. In order to count the number of vehicles, a virtual line is defined perpendicular to the road. Every time a centroid of a vehicle region cross the line, a counter C_T is increased. Moreover, when the motion direction indicates that the vehicle is moving far, a counter C_u is increased, and when the vehicle is coming to the camera C_d is incremented.

With these counters C_T, C_u, and C_d the algorithm can measure the number of vehicles that are using the road per unit of time (traffic mean velocity). With more counters we can count in addition to the number of vehicles that are moving on each direction, the number of vehicles that are moving on each lane (for instance, in a motorway).

When traffic jams are produced the mean traffic velocity decreases, but a decreasing of this mean velocity does not always indicate a traffic jam. To detect these facts we need to count the number of vehicles that are present in the image. A large number of vehicles and a little mean velocity indicates a traffic jam.

3.2 Detecting stopped vehicles

When a vehicle region has a velocity close to zero in a part of the image where usual speeds are quite different to zero and it is merged to the background for t continuous frames, this vehicle is considered as stopped. A stopped vehicle is never removed from the master list. Hence, the region of the object is always detected as a lost region and then recovered from the background. Motion parameters of a stopped vehicle are set to zero. Therefore its shape always remains stopped thanks to the information stored in previous frames until the vehicle moves again. This facility enables the system to count the number of vehicles that stop at the traffic lights when the light is red, and thereby, to detect immediately traffic problems at certain junctions or hard-shoulders.

3.3 Detecting anomalous events

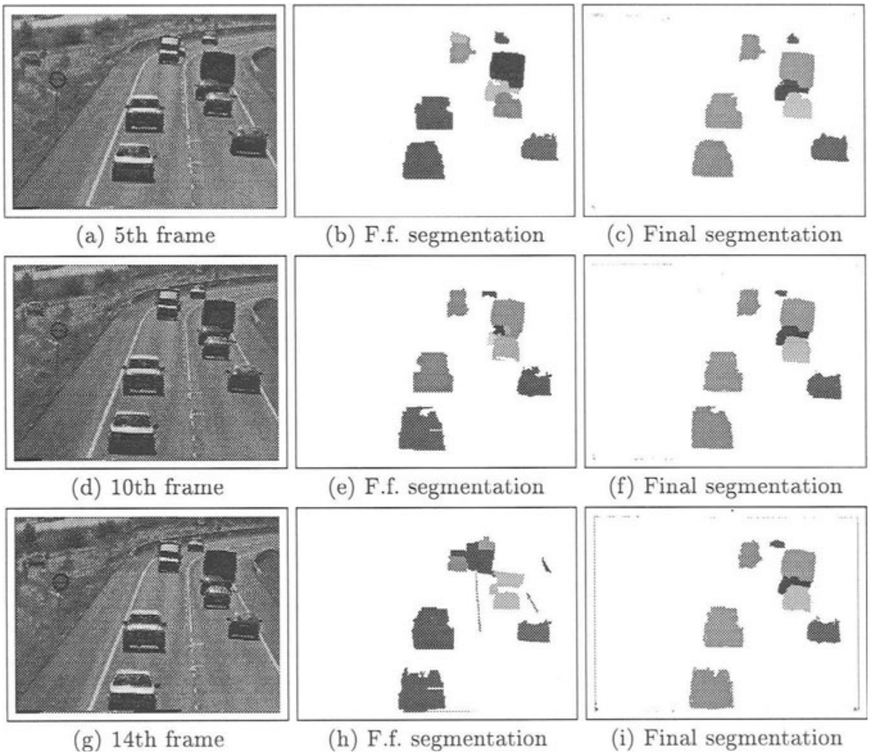
Dangerous situations such as accidents have to be detected by a traffic monitoring system and provoke an instantaneous warning. Our system can detect two types of accidents. On one hand, vehicles which cross danger lines. This virtual lines are defined in the system and delimit the part of the image on which the vehicles are moving. Danger lines are defined on the road boundaries or separating the two groups of lanes with different traffic direction in a motorway. When a vehicle cross a danger line, a warning is triggered which alerts of a dangerous situation. On the other hand, when two vehicles are moving with different directions and are only separated by a few pixels, it means that a vehicle crash is likely going to happen. Then another warning is produced which alerts to the traffic operator.

4 Results

This method has been tested with several traffic sequences. In figures 2 and 3 we present two of these sequences. First column of these figures presents several frames of the sequences. Second column contains the frame to frame segmentation of each frame, and third column is the final segmentation of the frame applying the whole algorithm, that is the result of integrating the segmentation information through several frames.

Figure 2 shows the segmentation of several frames of a sequence recorded at a road. These frames have been chosen due to their frame to frame segmentation presents some problems. Subfigure 2.a is the 5th frame of the sequence, and subfigure 2.b is its frame to frame segmentation. Here, the farthest truck has been splitted into two regions, but the final segmentation corrects this problem since in the previous segmentations the truck was formed by a single region.

In the segmentation of the 10th frame(subfigure 2.e), the intermediate car that is in the group of three vehicles is badly segmented. The region corresponding to this car was matched to the clearest region, which only contains part of the pixels of the car. The rest of pixels were assigned to the region of the car that


(a) 5th frame (b) F.f. segmentation (c) Final segmentation

(d) 10th frame (e) F.f. segmentation (f) Final segmentation

(g) 14th frame (h) F.f. segmentation (i) Final segmentation

Fig. 2. Road sequence. First column: original image. Second column: frame to frame segmentation. Third column: final segmentation of the frame

its covering it partially, and to a new region. However, the final segmentation (subfigure 2.f) is again right. The pixels are returned from both regions to the intermediate car region due to the fact that the values $M_{ij}^{R_i}.v$ of these pixels in its matrix are the largest.

The segmentation of the 14th frame (subfigure 2.h) is the worst of the sequence. In this segmentation some static regions have been detected as moving regions, part of the farthest truck has been lost, the first car and the second car of the group of three vehicles have been merged, and the truck of this group has disappeared. Moreover, the shape of some vehicles appears corrupted. In spite of these errors, the method corrects the segmentation. The static regions are removed and the vehicles recover their correct shape. The lost truck of the group is detected by the algorithm as a lost vehicle region, and their pixels are extracted from the background region.

Figure 3 shows a sequence in which two cars stop in a traffic lights. Frame to frame segmentation only can detect the first vehicle until the 37th frame(subfigure 3.b). In the rest of frames, the car is lost by frame to frame segmentation since

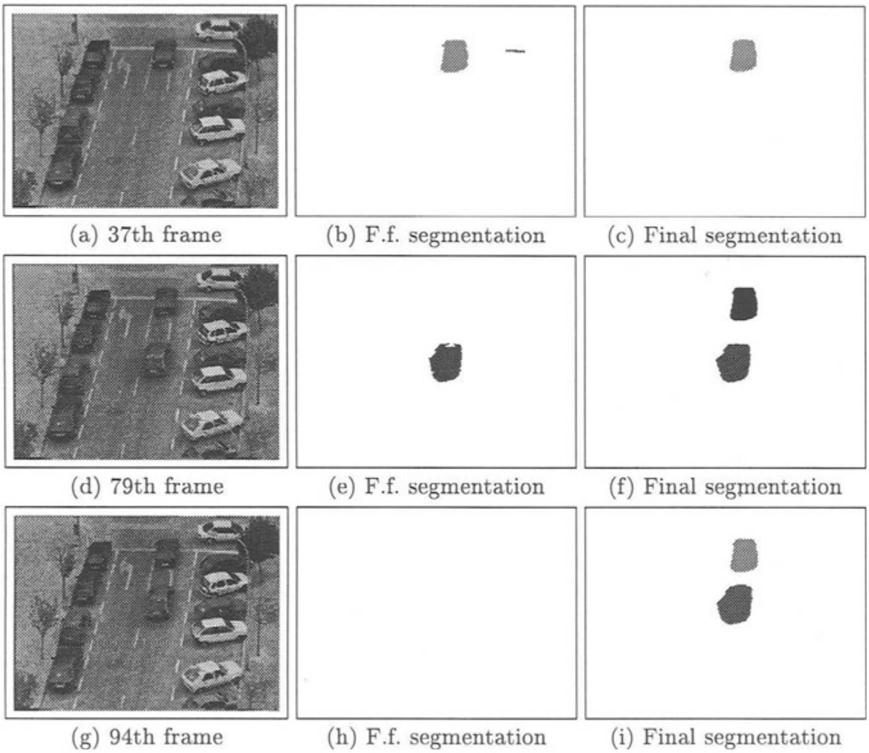

(a) 37th frame
(b) F.f. segmentation
(c) Final segmentation
(d) 79th frame
(e) F.f. segmentation
(f) Final segmentation
(g) 94th frame
(h) F.f. segmentation
(i) Final segmentation

Fig. 3. Traffic lights sequence. First column: original image. Second column: frame to frame segmentation. Third column: final segmentation of the frame

it stops. However, the final segmentation rightly segments it since it is declared by the algorithm as a stopped vehicle.

In 79th frame of traffic lights sequence a new car is moving into the image. This new vehicle is segmented by frame segmentation (subfigure 3.e), but, of course, not the other one. However, the final segmentation shows both vehicles.

In 94th frame (subfigure 3.g), both cars have stopped, and can not be detected by frame to frame segmentation. The algorithm detects their shape recognizing the vehicles as stopped. Both vehicles stay in the master region list, waiting for a frame in which their motions are detected by frame to frame segmentation.

5 Conclusions

In this work we have presented a new approach for segmenting and tracking moving objects on sequences. Our method is based on tracking regions and integrates segmentations provided by a frame to frame segmentation process, and accumulates the segmentation information for obtaining an improved segmentation. As opposed to usual methods, our method not only uses segmentation to achieve tracking of objects, but once tracking has been performed, it is used for improving segmentation.

The system provides useful facilities for traffic monitoring such as surveillance of vehicle trajectories, counting the number of vehicles that are using a road or a lane, counting the number of vehicles that have stopped at junctions or traffic lights, estimating the mean traffic velocity, or detecting traffic jams. Moreover, the system permits the detection of dangerous situations (accidents, stopped vehicles, vehicles that go off the road, etc.) that trigger an automatic warning.

References

1. R.C. Waterfall and K.W. Dickinson. Image procesing applied to traffic. *Traffic Engeneering Control*, 25:60 - 67, 1984.
2. M.P. Dubuisson and A.K. Jain. Contour extraction of moving objects in complex outdoor scenes. *International Journal of Computer Vision*, 14:83–105, 1995.
3. J. Badenas, M. Bober, and F. Pla. Motion and intensity-based segmentation and its application to traffic monitoring. In *Proceedings, International Conference on Image Analysis and Processing ICIAP'97, Florence, Italy*, pages 502–509. IAPR, 1997.
4. J.M. Ferryman, A. D. Worrall, G.D. Sullivan, and K.D. Baker. A generic deformable model for vehicle recognition. In *Proceedings of the British Machine Vision Conference*, pages 128–136. BMVA Press, 1995.
5. D. Koller, K. Daniilidis, and H.H. Nagel. Model-based object tracking in monocular image sequences of road traffic scenes. *International Journal of Computer Vision*, 10:257 -281, 1993.
6. D. Koller, J. Weber, and J. Malik. Robust multiple car tracking with occlusion reasoning. In Jan-Olof Eklundh, editor, *Proceedings, 5th European Conference on Computer Vision, (Berlin, 1994)*, pages 189–196. Springer-Verlag, 1994.
7. N.J. Ferrier, S.M. Rowe, and A. Blake. Real-time traffic monitoring. In *Proceedings, 2nd IEEE Workshop on Applications of Computer Vision*, pages 81–87, 1994.
8. P. McLauchlan and J. Malik. Vision for longitudinal vehicle control. In *Proceedings of the British Machine Vision Conference*, pages 260–269. BMVA Press, 1997.
9. S.M. Smith and J.M Brady. Asset-2- real-time motion segmentation and shape tracking. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17:814–820, 1995.
10. B.D. Steward, I. Reading, M.S. Thomson, T.D. Binnie, K.W. Dickinson, and C.L. Wan. Adaptative lane finding in road traffic image analysis. In *Proceedings 7Th International Conference on Real Traffic*, pages 133–136, 1994.

A Self-Diagnosing Distributed Monitoring System for Nuclear Power Plants

Aldo Franco Dragoni, Paolo Giorgini, Maurizio Panti

Istituto di Informatica, Università di Ancona
via Brecce Bianche, 60131, Ancona (Italy)
{dragon,giorgini}@inform.unian.it

Abstract. Sensor data fusion and interpretation, sensor failure detection, isolation and identification are extremely important activities for the safety of a nuclear power plant. In particular, they become critical in cases of conflicts among the data. If the monitored system's description model is correct and its components work properly, then incompatibilities among data may only be attributed to temporary deterioration or permanent breakage of one or more sensors. This paper introduces and discusses the conception of a distributed monitoring system able to attach each sensor with a statistically-evaluated relative degree of reliability, which is especially useful for devices situated in dangerous zones or areas, difficult to reach inside huge and complex power plants.

1 Introduction

Safety in nuclear power plants requires reliable information concerning the state of the process. Elaboration of data coming from the sensors of these complex plants is thus extremely important, and becomes critical when some sensors stop working properly. Failure detection, isolation and identification of sensor are indispensable activities for the monitoring system of a nuclear power plant. In recent years different approaches to the problem have been proposed. In [18], Kratz et al. presented a method based on the analytic redundancy for detecting and isolating sensor failures in a steam generator used in nuclear power plants. Keith and Belle in [19] developed signal validation software for application to nuclear power plants. Their system combines some previously established fault detection methodologies as well as some newly developed modules. Again Keith, in [20] presented a study on the feasibility of using a feed-forward backpropagation neural network in a signal fault detection capacity. The sensor data used in this study are taken from various subsystems within an operating nuclear power plant. More recently [21-23], Dorr and his colleague evaluated the contribution of analytic redundancy on the state estimation accuracy of linear systems. In these works, they carried out a comparative study of different methods of sensor fault detection using direct or indirect analytic redundancies on measurements obtained from a nuclear power plant.

Sensor Data Validation (SDV), gained much interest in the last few years [1,2,3,4]. As illustrated in [5], methods can be distinguished into three categories:

SDV1. *data*-based: they rely on statistical models obtained from observed data
SDV2. *model*-based: they rely on an analytical model of the monitored system
SDV3. *knowledge*-based: they rely on human expertise

Case 2 has been deeply studied in Artificial Intelligence, both as a *knowledge revision* problem (BR for short, see [6] for an overview) and as a *model-based diagnostic* problem (MBD for short, see [7] for a survey). It seems evident to us that BR and MBD are dual problems. In the last decade, MBD moved from its theoretical foundation [8][9] to some practical applications (see for instance [10]). In MBD, diagnoses are found from discrepancies between observation and prediction. The intermediate step is the exhaustive generation of the "conflict sets" for the tuple (*SD*,*COMPS*,*OBS*), in which *S*ystem *D*escription and *OBS*ervations are sets of first-order sentences, *COMP*onent*S* is a finite set of constants each one representing a component of the system [11]. A diagnosis is a subset of *COMPS* that covers all the conflict sets.

A problem with MBD is that each of its three fundamental steps, prediction, conflict recognition and candidate generation, exhibits a combinatorial explosion for large devices [12]. However, the worst problem with MBD is that it is difficult to find out a correct model for the system to diagnose. This paper does not deal with these problems: both of them will be cravenly avoided by imposing the relative simplicity of the apparatus to be controlled or diagnosed. Instead, this paper regards three ides:

1. the problem of recognizing *sensors'* faults can be approached entirely within the framework of MBD
2. from the diagnostics of the sensors' faults one can formulate interesting conclusions regarding the various sensors' relative reliability
3. from the estimated reliability of the sensors one can hypothesize the actual state of the monitored physical system even in cases of not-redundant conflicting data.

Normally, sensors come labeled with many important qualifications (accuracy, average life-time, ...) which are necessary to estimate their *a priori* current reliability. By "reliability" of a sensor we mean the "*probability* that the sensor is providing the correct measure," whatever the term "correct" may signify. However, the *actual* current reliability of a sensor may be lower than the "a priori" one due to unpredictable and/or unknown events that might have affected it from its assembly to its current employment in the monitoring system. Of course, any sensor's current conditions can be appraised through appropriate testing devices (actually, the maintenance of a nuclear power plant's monitoring system consists of systematic controls and calibration of sensors during the annual shutdown of the plant). But, apart from the academic problem of infinite regression (which devices will test the testing devices, and so on, ...), a concrete question is that "testing" has its own costs. For instance, in the monitoring apparatus of an automatic production line, some optical sensors might have been altered after a temporary fault of the conditioning device that cleans the air from the pollution particles produced by the power generator. Since testing the sensors implies stopping the manufacturing process, other evidence about their possible deterioration would be appreciated. Furthermore, systematic controls and calibration of sensors during the annual shutdown of the plant "can lead to material degradation due to repetitive manipulations" [27]. In the case of thermic sensors situated in proximity to the kernel of a nuclear power plant, evidence about the wearing away of a limited number of sensors will drastically reduce the costs of maintenance. Issue 2, above, suggests that such an evidence may come from the theoretical model of the monitored process/phenomenon and from the *global* datum provided by the *distributed* monitoring apparatus.

The *group* of sensors acts as a testing device for each one of its own members. Of course, this evaluation depends on the average reliability of all the sensors in the group (hence including the corrupted ones) and on the accuracy of the monitored entity's model. In any case, these estimates will not be comparable (nor for quality neither for typology) with the evaluations made by specifically designed testing devices. Their point is that they do not interfere in any way with the manufacturing process, thus they have no expenses at all (apart from the fixed costs of a CPU, some data-acquisition boards and a mass storage device). A key idea with this distributed auto-estimate is that of "minimal conflicts". Intuitively, if it has been detected a minimal conflict between the sensors A and B (by confronting the collected data with the theoretical model) and, subsequently, another minimal incompatibility is found involving B and C, then one may suppose more probable the deterioration of B than those of both, A and C. Dealing with probabilities, we do not want to reinvent the wheel since Bayesian Conditioning seems an appropriate tool to accomplish the task. Basically, the new reliability of a sensor S will be computed as the probability that S gave the correct value provided that it has been involved in some minimal conflicts. The greater the cardinality of these minimal conflicts, the higher the chance that S is working properly. The worst case is when S in involved in a singleton minimal conflict (i.e., it went, by itself, out of the range compatible with the theoretical model) so that its new reliability is 0. We will estimate statistically the current reliability of each sensor (over all its working life) w.r.t. the other ones.

There are cases in which the cost of testing a sensor is infinite, i.e., the examination is impossible or not convenient. Let us think about the sensor equipment of unmanned satellite stations or about real-time domains in which you receive impossible (or utterly improbable) global data and have no time to test the sensors. These cases fall into the classic discipline of *decision support under uncertainty.* In these circumstances, the estimated *current* ranking of reliability plays an important role since, although very rough, it provides a more justified and up to date (hence more adequate) estimate than the "a priori" one. To accomplish this task, the fundamental tool we adopted in our method is Dempster's Rule of Combination in the special form in which Shafer and Srivastava apply it to the "auditing" domain [14].

2 Diagnosing sensor faults

Diagnosing sensor faults can be done within the framework of MBD [9] by extending the system's description (e.g., figure 1-A) to encompass the sensors' models (e.g., figure 1-B).

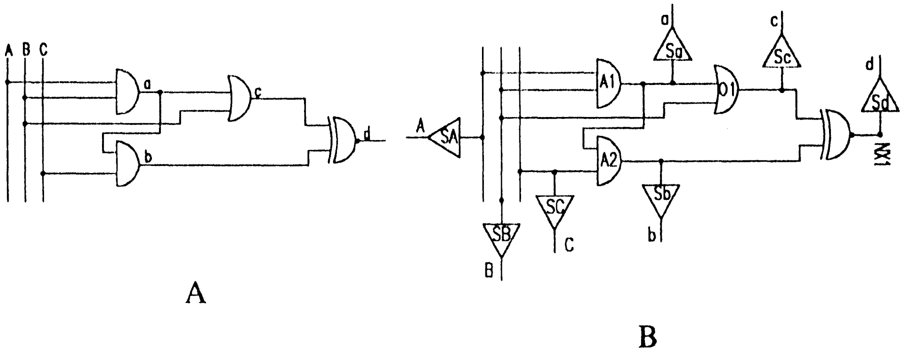


Fig. 1. Extending the notion of system to encompass the sensors models

The system's description will be extended congruously (in bold below):

COMPS :{ A1, A2, O1, NX1, **S_A, S_B, S_C, S_a, S_b, S_c, S_d** }

SD : ANDG(x) $\wedge$ $\neg$AB(x) $\Rightarrow$ out(x)=and(in1(x),in2(x))
NXORG(x) $\wedge$ $\neg$AB(x) $\Rightarrow$ out(x)=nxor(in1(x),in2(x))
ORG(x) $\wedge$ $\neg$AB(x) $\Rightarrow$ out(x)=or(in1(x),in2(x))
SENS(x) $\wedge$ $\neg$AB(x) $\Rightarrow$ out(x)=in(x)
ANDG(A1), ANDG(A2), NXORG(NX1), ORG(O1)
SENS(S_A), SENS(S_B), SENS(S_C), SENS(S_a), SENS(S_b), SENS(S_c), SENS(S_d)
out(A1)=in1(O1), out(A1)=in1(A2), out(A2)=in2(NX1), out(O1)=in1(NX1)
in2(A1)=in2(O1), in(S_A)=IN1(A1), in(S_B)=IN2(A1), in(S_C)=IN2(A2),
in(S_a)=OUT(A1), in(S_b)=OUT(A2), in(S_c)=OUT(O1), in(S_d)=OUT(NX1)
in1(A1)=0 $\vee$ in1(A1)=1, in2(A1)=0 $\vee$ in2(A1)=1, in2(A2)=0 $\vee$ in2(A2)=1

OBS : a finite set of first order ground sentences

The system components COMPS is a finite set of constants each one representing a component of the system, sensors included. The system description SD describes how the system components *normally* behave by appealing to the distinguished predicate AB whose intended meaning is "abnormal". Thus, the first sentence states that a normal (i.e. not ABnormal) and gate's output is the Boolean and function of its two inputs. Recalling from [9], a *minimal conflict set* for (*SD*,*COMPS*,*OBS*) is a subset $\{x_1,\dots,x_k\}$ of *COMPS* such that $SD \cup OBS \cup \{\neg AB(x_1),\dots,\neg AB(x_k)\}$ is inconsistent and such that the same holds for no proper *sub*set of $\{x_1,\dots,x_k\}$. Any *minimal hitting set* on the collection of all the minimal conflict sets will be a *diagnosis* for (*SD*,*COMPS*,*OBS*). The strength of this framework is its ability to diagnose the contemporary faults of components and sensors. However, often sensors observe physical systems in which there is no notion of component at all. In these cases, *COMPS* contains only the sensors, *SD* reduces to a mathematical model (maybe very complex) of the observed phenomenon and *OBS* is a simple array of numerical and/or boolean data. As an example, let us consider a metallic bar, heated at an extremity and monitored by some thermometers as depicted in figure 2.

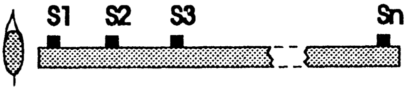


Fig. 2. Diagnosing faults of pure sensor systems

Even ignoring the bar's heat transfer equation, we may yet model the system with simple constraints (in bold face below for the case of only three thermometers):

COMPS: $\{S_1, S_2, S_3\}$
SD : $\mathrm{SENSOR}(x) \wedge \neg \mathrm{AB}(x) \Rightarrow \mathrm{out}(x)=\mathrm{in}(x)$, $\mathrm{SENS}(S_1)$, $\mathrm{SENS}(S_2)$, $\mathrm{SENS}(S_3)$
$\mathbf{out}(S_1) \geq \mathbf{out}(S_2)$, $\mathbf{out}(S_2) \geq \mathbf{out}(S_3)$
OBS : a triple of numerical data

The strongest point with the adoption of MBD in SDV relies in the notion of *good* (as we called it for the obvious duality with de Kleer's *nogood*, called "minimal conflict set" by Reiter), that is a subset $\{x_1,\ldots,x_k\}$ of *COMPS* such that $SD \cup OBS \cup \{\neg AB(x_1),\ldots,\neg AB(x_k)\}$ is *consistent* and such that the same holds for no proper *super*set of $\{x_1,\ldots,x_k\}$. Each good is the complement of a diagnosis w.r.t. *COMPS*, i.e., a maximally consistent set of sensors. Goods play an important role when trying to hypothesize the system's status in presence of conflicting data. In fact, because of the duality between goods and diagnoses, choosing a most probable diagnosis means choosing a most probable good, i.e., a most probable (and complete) reconstruction of the system's status.

A problem with MBD applied to SDV is that, independently of the accuracy of *SD*, the theory $SD \cup OBS \cup \{\neg AB(x) \mid x \in COMPS\}$ may be *consistent* even in cases of sensor faults. These *hidden faults* may occur, for instance, in cases of contemporary breakage of more than one sensor such that the global output is still a possible (although wrong) one.

3 Estimating the sensors reliability and the goods credibility

Whereas hidden faults constitute a problem, successful recognition of *minimal* conflicts offers an invaluable opportunity to estimate, statistically, the actual current sensors' reliability from the "a priori" one. The most obvious way to do this is through Bayesian Conditioning, since we defined "sensor's reliability" as the *probability* that the sensor is returning the correct value. Let us denote with R_i and NR_i, respectively, the "a priori" and the "a posteriori" reliability of the sensor S_i, and let us denote with S the set *COMPS* restricted to the sensors. Under the assumption that the deterioration of each sensor is an independent event (!?!), the hypothesis that only those belonging to $\Phi \subseteq S$ are working properly has the combined "a priori" probability $R(\Phi) = \prod_{S_i \subset \Phi} R_i \cdot \prod_{S_i \notin \Phi} (1 - R_i)$. It holds that $\sum_{\Phi \in 2^S} R(\Phi) = 1$. Of course, after the recognition of a minimal conflict ϕ, $NR(\Phi)=0$ for each $\Phi \supseteq \phi$, and any other Φ is subjected to Bayesian Conditioning so that $\sum_{\Phi \in 2^S \wedge \phi \not\subset \Phi} NR(\Phi) = 1$. The "a posteriori" reliability of S_i is defined as $NR_i = \sum_{S_i \in \Phi} NR(\Phi)$. If S_i is involved in *minimal* conflicts, then $NR_i \leq R_i$, otherwise $NR_i = R_i$. Estimating the current reliability CR_i of a sensor S_i from R_i and from the history of the recognized minimal conflicts is a (debatable) statistical matter. In the experiments below, we took for CR_i the average of all the NR_i calculated during the life of the distributed monitoring system. As we'll see, such a CR_i provides an interesting relative ordering of reliability. The overall distributed sensor system acts as a testing device for each of its constituent member.

Note that NR_i is calculated only on the reception of conflicting data. Another important question is that of the length of the temporal window, i.e., how far we go back in the past to record conflicting data; intuitively, the wider the window the higher the inertia of the mechanism in registering the sensors' deterioration.

In Shafer's and Srivastava's multi-source version of the belief function framework [14], the sources' degrees of reliability are "translated" into belief-function values on the given pieces of information. In our method we follow them by taking the estimated reliability CR_i as primary *evidence* in favor of the datum s_i furnished by S_i. Let Ω denote the set of all the possible configurations of the monitored system, and let $[s_i] \subseteq \Omega$ denote only those compatible with s_i. The key assumption is that *a reliable sensor cannot give false information, while an unreliable sensor can give correct data*; the hypothesis that "S_i is reliable" is compatible only with $[s_i]$, while the hypothesis that "S_i is *un*reliable" is compatible with the entire Ω. Each S_i gives an evidence for Ω and generates the following *basic probability assignment* (*bpa*) m_i over the elements X of 2^Ω:

$$m_i(X) = \begin{cases} CR_i & \text{if } X = [s_i] \\ 1 - CR_i & \text{if } X = \Omega \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

All these *bpa*s will be then combined through Dempster's Rule of Combination (DRC):

$$m(X) = m_1(X) \otimes \ldots \otimes m_n(X) = \frac{\sum_{X_1 \cap \ldots \cap Xn = B} m_1(X_1) \cdot \ldots \cdot m_n(X_n)}{\sum_{X_1 \cap \ldots \cap Xn \neq \varnothing} m_1(X_1) \cdot \ldots \cdot m_n(X_n)} \tag{2}$$

From the combined *bpa* *m*, the credibility of a set of data (hence of a good) s is given by

$$Bel(s) = \sum_{X \subseteq [s]} m(X) \tag{3}$$

A major problem with the belief function formalism is the computational complexity of DRC; the straightforward application of the rule is exponential in the cardinality of Ω and in the number of sensors. However, much effort has been spent in reducing its complexity. Such methods range from "efficient implementations" [15] to "qualitative approaches" [16] through "approximate techniques" [17].

4 Experiments

We made a series of simulation experiments aimed at value the *sensibility* of the proposed mechanism (capacity to distinguish small differences among sensors' reliabilities) and its *robustness* (capacity to perform acceptably in very degraded situations). The simulator takes in input the sensors' real degrees of capacity (C_i), their degrees of "a priori" reliability (R_i), the model R of the monitored system and the length of the simulation. At any cycle it:

1. simulates a correct data acquisition
2. simulates an error (it alters the datum of each S_i according to C_i taken as a fault frequency)
3. if the resulting global data contradicts with R it starts the proposed mechanism.

We made several experiments with the heated bar of figure 2 modeled as described in section 2. Trying to evaluate the system's sensibility, we ran some simulations with only one deteriorated sensor. We recapitulate here the main results:

- the system is always able to find the corrupted sensor almost immediately independently of its capacity
- the time necessary to find out the corrupted sensor grows with its capacity
- these results do not depend on the sensors' output range (probably, this is due to the very particular error typology we considerate)
- results improve with the number of the sensors (more sensors, better results).

Having found always the corrupted sensor, the mechanism has been invariably able to recognize the correct maximal consistent set of data through the DRC. We made other simulations with two or more deteriorated sensors. We can say that:

- the mechanism is still able to find the corrupted sensors
- the estimated degrees of reliability of the corrupted sensors are close to those of the properly working ones
- the mechanism needs more time to find the corrupted sensors
- correct sensors are innocently involved in more minimal conflicts so that their degrees of reliability decrease.

We also simulated the monitoring system for the digital circuit in figure 1.B. Again, we tried some simulations with only one deteriorated sensor (with a capacity of 0.5). We can say that:

- the mechanism is still always able to find out the corrupted sensor almost immediately independently of its reliability
- the time necessary to find out the corrupted sensor grows with its reliability
- the mechanism finds the corrupted sensor but its esteemed reliability depends on the particular sensor.

Again, having found always the corrupted sensor, the system chooses always the correct maximal consistent set of data.

To evaluate the robustness we ran some simulations with two or more sensors with reliability 0.5. We can say that there is a dependence of the system's performances from the particular set of corrupted sensors. We called this effect *overexposure*: by overexposure of a sensor we mean its higher probability of being involved in minimal conflicts. We tried to find some correction factors to limit the impact of this ugly effect; the problem is that such factors are not fixed numbers but are function of the unknown real capacity of the sensors.

5 Related work and conclusions

The system presented in this paper belongs to both the SVD1 and SVD2 category of sensor data validation. The model-based methods presuppose the existence of an analytic model of the system and are based on a common methodology: generating and analyzing of the signals sensitive to the fault *residues* [24]. The main approaches to generate the residues can be classified in approaches based on *analytic redundancy* [27] (e.g. *parity space*) and approaches based on the *observator* (e.g. *Kalman filter*). In Dorr et al. [21-23] faults are detected by comparing each residual with thresholds defined with respect to the sensor measurement accuracy. Residuals are generated by comparing each measurement with an estimate value. They can be obtained by *simple redundancy* (at least two sensors that measure the same physical variable) or *analytical redundancy*. The model-based methods are very efficient when there is a linear and well-known model of the system. The problem has been also deeply studied in Artificial Intelligence. In [2] Lee presents a technique based on the analytic redundancy that needs an accurate knowledge of the process. Also Washio et al. [25] propose a method to find the sensors' faults based on the model of the monitored process. Detection of faults is based on consistency checking between observations and optimal constraints, called "minimal over-constraints", consisting of first principles. If some inconsistencies are detected, the model-based diagnosis is applied to derive the candidates of faulty components. A model consists of constraints under the normal condition of the component including its sensors. The individuation of the sensors' faults and the diagnosis of the components are performed contextually in the same framework. In particular, the method allows diagnosing non linear components, sensors and components, and the width of the fault.

Our method exploits any kind of knowledge to extract the *minimal conflicts*, allowing to use both equations and constraints of real situations like "if the temperature is higher than 100° C, then the alarm starts". In the SVD, it is supposed that the components are not corrupted. In our approach this constraint can be overcome extending properly the method. The historical analysis of the data allows exploiting information formerly draw out for solve the future conflicting situations. LES and the systems proposed in [18] don't give indications concerning how to solve the conflicts and how to choose one of the possible diagnoses as our does.

The hardest problem with our method is what we called "overexposure effect". We believe that effective solutions to this problem depend on the particular application domain

	Dorr et al.	Washio et al.	Our
Method of diagnosis	Through standardized least square residuals (SLSR) analysis	Through minimal conflicts	Through minimal conflicts
Multiple faults of components		•	•
Multiple faults of sensors	•	•	•
Non-linear models	•	•	•
Highly non-linear models		•	•
Dynamic behaviors		•	•
Faults amplitudes	•	•	
Disequations in System Description			•
Estimated reliability of the sensors			•
Identify anomalous quantities directly disturbed by faults		•	
Complexity	Iterative method	Exp. in the number of undetermined quantities in SD	Exp. in the cardinality of Ω

Table 1.

Acknowledgement

We'd like to thank our student, Marco Pandolfi, who compared our approach with those reported in table 3.

References

1 Alag S. and A. M. Agogino, "A Methodology for Intelligent Sensor Validation, Fusion and Sensor Fault Detection for Dynamic Systems" #95-0301-P, 1995.

2 S. C. Lee, "Sensor Value Validation Based on Systematic Exploration of the Sensor Redundancy for Fault Diagnosis KBS", IEEE Transactions on Systems, Man, and Cybernetics, Vol 24, n° 4, April 1994.

3 S. Iksch, W. Horn, C. Popow, F. Paky, "Context-sensitive data validation and data abstraction for knowledge-based monitoring", in ECAI 94. 11th European Conference on Artificial Intelligence, 1994, pp. 48-52.

4 D. Himmelblau and M. Bhalodia, "On-Line Sensor Validation of Single Sensor Using Artificial Neural Networks", in Proceedings of the 1995 American Control Conference, vol. 1, 1995, pp 766-770.

5 Stephane Canu, "Sensor Data Validation: a Review", in EM2S Esprit European Project.

6 Dragoni A.F., Belief Revision: from theory to practice, "The Knowledge Engineering Review", vol 12, n° 2, Cambridge University Press, 1997.

7 de Kleer, J., Hamsher, W. C., Console, L., (Eds.): Readings in Model-Based Diagnosis, Morgan Kaufmann, 1992.

8 de Kleer, B., C., Williams, Diagnosing Multiple Faults, Artificial Intelligence 32, pp. 97-130, 1987.

9 R. Reiter, A Theory of Diagnosis from First Principles, Artificial Intelligence 32, pp. 57-95, 1987.

10 Beschta, A., Dressler, O., Freitag, H., Montag, M., Stru§, P., A model-based approach to fault localisation in power transmission networks, Intelligent Systems Engineering, 2(1), IEE Publisher, pp. 3-14, 1993.

11 de Kleer, J., Mackworth, A. K., Reiter, R., Characterizing Diagnoses, Proceedings of 1990 Conference of the American Association for Artificial Intelligence, pp 324-330, 1990.

12 Johan de Kleer, Focusing on Probable Diagnoses, Proceedings of 1991 Conference of the American Association for Artificial Intelligence, pp 842-848, 1991.

14 Shafer G. and Srivastava R., The Bayesian and Belief-Function Formalisms a General Perpsective for Auditing, in G. Shafer and J. Pearl (eds.), Readings in Uncertain Reasoning, Morgan Kaufmann Publishers, 1990.

15 Kennes, R., Computational Aspects of the Möbius Transform of a Graph, IEEE Transactions in Systems, Man and Cybernetics, 22, pp 201-223, 1992.

16 Parson, S., Some qualitative approaches to applying the Demster-Shafer theory, Information and Decision Technologies, 19, pp 321-337, 1994.

17 Moral, S. and Wilson, N., Importance Sampling Monte-Carlo Algorithms for the Calculation of Dempster-Shafer Belief, Proceeding of IPMU'96, Granada, 1996.

18 Kratz, F., Mourot, G., Maquin D., Loisy F., Detection of measurements errors in nuclear power plants, in IFAC Symposia Series n 9 1992. Publ Pergamon Press Inc, Tarrytown, NY, USA. p 165-170.

19 Keith E. H., Belle R.U., An Integrated Signal Validation for Nuclear Power Plants, Nuclear Technology, vol. 92 N. 3, pp 411-427, 1990.

20 Keith E.H., Neural Networks for Signal Validation in Nuclear Power Plants, Proceedings of the Second Annual Industrial Partnership Program Conference, March 9, 1992.

21 Dorr R., Mourot F., Kratz F., Loisy F., An application of gross errors measurements detection in nuclear power plants, IFAC Safeprocess '94, June 1994, Espoo, Finland.

22 Dorr R., Kratz F., Ragot J., Loisy F., Germain J.L., Sensor validation in a nuclear power plant using direct and analytical redundancy, International Conference on Industrial Automation, June 1995, Nancy France.

23 Dorr R., Kratz F., Ragot J., Loisy F., Loisy F., and Germain J., Detection, Isolation, and Identification of Sensor Faults in Nuclear Power Plants, IEEE Transaction on Control Systems Technology, Vol. 5, no. 1, January 1997.

24 Patton R.J., Franck P.M., and Clark R.N., Fault Diagnosis in Dynamic Systems: Theory and Application. Englewood Cliffs, NJ: Prentice-Hall, Series in systems and Control Engineering, 1989.

25 Washio, Sakuma, Kitamura. "A new approch to quantitative and credible diagnosis for multiple faults of components and sensors". Artificial Intelligence Vol 91 pp. 103-130.

Overload Screening of Transmission Systems Using Neural Networks

J. Riquelme[1], A. Gómez[1], J.L. Martínez[1] and J.A. Peças Lopes[2]

[1] Department of Electrical Engineering, University of Seville
Escuela Superior de Ingenieros
Camino de los Descubrimientos, s/n
41092 Seville, Spain

[2] INESC, Instituto de Engenharia de Sistemas e Computadores
Largo Mompilher 22
4433 Porto, Portugal

Abstract. The process of determining whether a power system is in a secure or insecure state is a crucial task which must be addressed on-line in any Energy Management System. In this paper, an Artificial Neural Network, capable of accurately identifying the set of harmful contingencies, is presented, along with several results obtained from a real-size power network. The proposed approach makes use of classical numerical techniques to compensate the ANN's inputs so that it can deal with topological changes in the power system.

1 Introduction

A reliable power system should be operated in a "secure" state, which implies that the load is satisfied and no operating limits are violated under present operating conditions, but also after the occurrence of several unforeseen contingencies (e.g., outages of lines, transformers or generators)[13].

The process of determining whether the power system is *secure* or *insecure* is known as "Static Security Assessment", whose key issues are [2]:

1. First, a fast identification of the set of dangerous critical contingencies must be performed ("Contingency Selection or Screening").
2. Next, the impact of the critical contingencies on the power system must be evaluated, usually through a post-contingency "Load Flow" routine ("Contingency Analysis").

Given this information, a system operator can decide if preventive actions should be initiated to mitigate the potential problems.

A good "Contingency Selection & Analysis" package must be computationally fast and accurate in correctly identifying the harmful contingencies, that is, quickly reducing the number of contingencies to be studied in detail by eliminating all the non-critical ones.

Several algorithmic techniques have been proposed, most of which are currently being used in Energy Management Centers, to reduce the number of critical contingencies to be studied explicitly in real-time, the most popular being "Distribution Factors" [12, 9], "Performance Index" [6], and "Approximated Load-Flow" [15, 3, 4, 1] methods. In reference [11], a performance evaluation of several classical static security algorithms is presented.

Over the past few years, several approaches using "Artificial Neural Networks" (ANN) have been proposed as alternative methods for "Static Security Assessment" in power system management, both using supervised and unsupervised architectures [7, 10]. However, both classical and ANN-based methods must deal with the following problems:

- The number of possible system configurations (i.e., network topologies) is large.
- The state of every system configuration is typically defined by several hundreds or thousands of line/transformer power flows, along with a similar number of electrical bus voltages.
- Each pre-contingency configuration gives rise to a huge number of post-contingency topologies.

Perhaps owing to the combinatorial explosion of the number of contingencies and variables, almost all ANN applications to "Security Assessment" reported in the literature have only been applied to small academic power systems.

In this article, an ANN intended to carry out a "Contingency Screening" for line/transformer overloads is presented and tested on the *Andalusian Transmission Network*, composed of about 130 electrical buses (400 and 220 kV buses) and 250 lines and transformers.

The paper is structured as follows: First, an ANN for a specific system topology is presented, following the techniques proposed in [5]. Secondly, a hybrid approach, which makes use of an ANN along with classical numerical techniques, is also presented and tested, showing that it is capable of dealing with several pre-contingency topologies by "compensating" the input variables of the ANN.

2 Scheme 1: ANN Based on Line Power Flows

The power system for which the NN is intended refers to the Southern Spanish Transmission Network, comprising 256 branches (lines/transformers), 130 buses and 22 generators (these figures include an equivalent of the external national network).

The ANN adopted is of the multi-layer perceptron (MLP) type, its structure being determined by the following considerations:

- As the goal of the ANN is to detect line overloads, it seems intuitive to choose the line apparent power flows as input variables. This information is readyly available as a byproduct of the state estimator. However, in order to reduce the number of inputs as much as possible, a feature selection algorithm,

based on the $\mathcal{F}$ index [8, chapter 5], was used to determine which of the 256 possible power flows were actually significant. As a result, only 54 branch power flows were selected.
- There is no clear guidelines to decide how many neurons should comprise the hidden layer. In our case, after several trials, it was decided that 108 hidden neurons constitute a good compromise between complexity and performance.
- Ideally, the ANN should have as many output variables as relevant outages we wish to assess. In practice, however, operators are much more interested in the consequences of outages affecting the 400 kV transmission lines, because they carry huge amounts of power. In this case, 7 out of the 11 400 kV internal branches, plus 3 branches of the external network, were monitored. In the end, only 9 output variables were adopted, since it was noticed that two of the contingencies systematically led to identical results. The output values are constrained to the interval $[0, 1]$, a value larger than 0.5 indicating that the outage associated with this particular output causes overloading in any system component.

This MLP, whose neurons make use of the sigmoid transfer function, was trained by means of the back propagation algorithm. As many as 746 network states were generated by systematically modifying the load profile and running a load flow program. Note that all situations refer to the same base-case topology. Two thirds of these states were chosen to train the NN, keeping in mind that both secure and insecure states were included in the same proportion, while the remaining subset was used in the test phase.

The performance of the MLP on these test cases was extremely good. There were no missed detections and only a single false alarm, which represents an error of 0.4% over the entire test set (0.04% when the 10 outputs per case are considered). Since the main purpose of the ANN is to detect critical contingencies, which should be subsequently analyzed in detail by a load flow routine, the existence of a very small percentage of false alarms is not too important, because the consequence of this kind of error is merely a moderate increase in the computational cost at the contingency analysis stage.

Although most of the time all components of the transmission network are in service in order to increase the security level, it sometimes happens that a particular line or transformer is disconnected owing to periodical maintenance, etc. The ANN described above was also tested using a battery of cases in which the base-case topology was modified by disconnecting a single element. According to expectations, the percentage of misclassifications largely exceeded 30%, because the NN had not been trained with the modified topologies.

Consequently, a new set of 5861 cases was created comprising 5 different topologies in addition to the base-case topology. Four of the new topologies differ from the base case in a single element, while the fifth case constitutes a rather extreme case in which 3 components are simultaneously disconnected. When 2/3 of the whole set of cases are used again to train the NN, only 12 out of the 1953 test cases led to one or several classification errors, as shown in Table 1.

Topology	0		1		3			5				
Case #	1	2	3	4	5	6	7	8	9	10	11	12
False Alarms	1	1	1	1	1	2	1	0	0	2	1	1
Missed Detections	0	0	1	1	0	0	0	6	6	0	0	0

Table 1. Statistics corresponding to the 12 cases containing one or several wrong outputs.

The following comments are in order regarding these results:

- The NN gives now 2 false alarms, instead of 1, for the base-case topology (labeled 0 in Table 1). This performance deterioration is caused by the new learning process, aimed at tuning the NN to 5 additional topologies.
- The 2 missed detections corresponding to topology #1 could have been avoided with a more conservative threshold (the value of the 2 affected outputs was 0.48 while the threshold was 0.5). However, this solution would lead to extra false alarms.
- As expected, the worst results were obtained with the fifth topology, which significantly departs from the base case (3 elements are disconnected). No errors appear with topologies 2 and 4.
- Overall, taking into account that each of the 1953 test cases provides information about 10 different contingencies, the 12 false alarms and the 14 missed detections summarized in Table 1 represent only 0.06% and 0.07% respectively of the whole set of NN outputs. This performance is better than the one obtained with conventional contingency screening techniques, based on distribution factors or incremental load-flow models [11].

3 Scheme 2: ANN Based on Bus Power Injections

Although the results presented in the previous section are promising, it should be remembered that the NN was trained by means of only 6 different topologies to detect the overloads caused by just 10 contingencies. It would be quite difficult, if not impossible, to develop a single NN capable of recognizing any conceivable network topology and providing an output for every branch and generator outage, at least for realistic networks.

In an attempt to somehow elude the dependence on network topology and on the particular contingency being studied, a NN based on bus power injections has been developed and tested. As discussed in [8, chapter 9], power flows contain implicit information on the topology of the network, but the dimension of the power injection vector is smaller and the absence of input redundancy is guaranteed.

The basic idea behind the new NN is that both what makes the current topology different from the base case and the information on the particular contingency being studied, can be approximately represented by properly modifying the injection vector, much in the same way as compensation techiques [14] are applied in conventional methods to deal with localized network changes.

Again, the MLP type was adopted but, according to the new operating principles, its structure is as follows:

- The output layer comprises a single neuron whose output provides the same information as before, i.e., a value higher than 0.5 indicates that at least one branch is overloaded. This means that the NN must be resorted to every time a particular contingency has to be studied, unlike in the previous scheme where a single pass provided all outputs corresponding to the reduced set of contingencies for which the NN had been trained.
- The input buffer, as stated earlier, contains as many elements as the number of buses, plus an extra element representing the total demand. The precise meaning of these inputs will be explained below.
- The number of neurons comprising the hidden layer has been chosen in acordance with the number of network branches. A similar choice was made in [8], where 36 neurons were found optimal for a 35-branch network.

This time, the transfer function adopted is the hyperbolic tangent and the learning process is based on a modification of the cumulative back propagation rule known as the "Normalized-Cumulative-Delta Learning Rule".

The values fed into the NN inputs depend on the case in hand. Both in the learning and test phases the NN is used as if the network topology was always that of the base case. Consequently, when we are dealing with any base-case load profile, the NN is trained and tested by simply setting every input value to the net active power injection of the respective bus, normalized with the total demand. If we are interested in the outage of a branch carrying a certain active power flow from bus i to j, this flow must first be added to and substracted from the injection at buses i and j respectively, before resorting to the NN. This modification of the involved power injections is not intended to accurately compensate for the outaged branch, but as a simple means of informing the NN that there has been a change with respect to the base case affecting the two buses. When the topology of the current network state differs from that of the base case, the missing element is first identified (owing to security reasons, it is very unlikely that several components are simultaneously disconnected at the high-voltage levels). Then, a load flow program is run to find the power flows that would exist if the disconnected element was in service (the overhead caused by this numerical step is still modest compared to the cost of the overall process). From this step on, the way of proceeding is similar to the one explained above, except for the fact that the adjacent injections of both the voluntarily disconnected and the outaged branches must be properly modified.

The role played by the ANN in the contingency selection phase of the whole security assessment process is shown in figure 1.

In order to assess the efectiveness of this alternative approach, the same transmission network of section 2 has been used. In this case, the network is composed of 123 buses and 227 branches, and the total load amounts to 5711

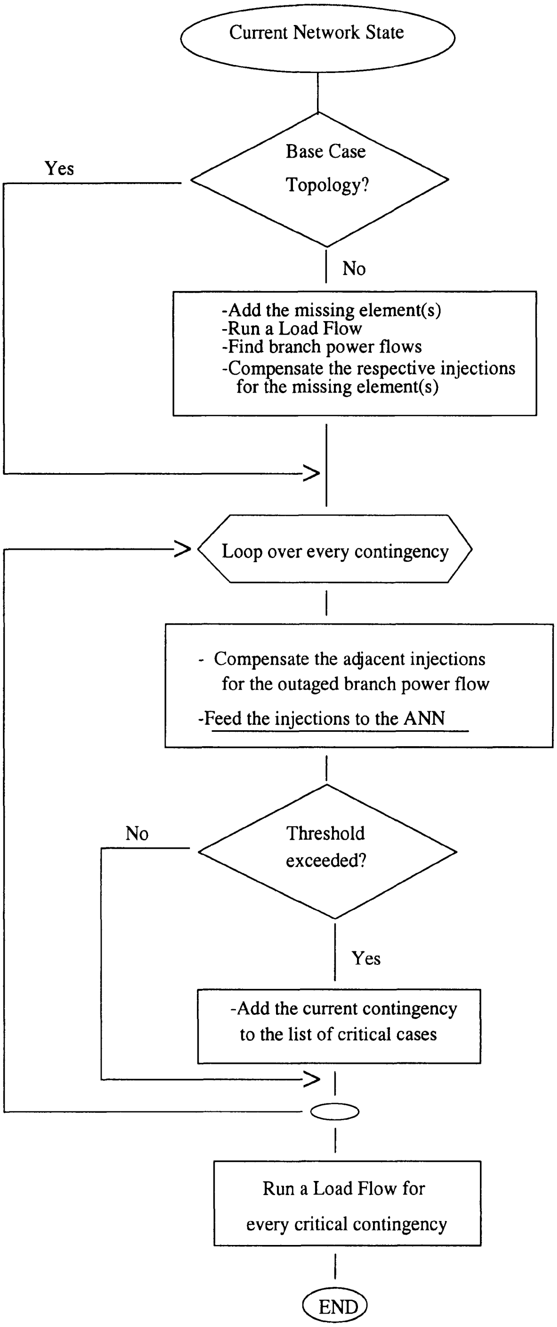

Current Network State
Base Case Topology?
Yes
No
-Add the missing element(s)
-Run a Load Flow
-Find branch power flows
-Compensate the respective injections for the missing element(s)
Loop over every contingency
- Compensate the adjacent injections for the outaged branch power flow
-Feed the injections to the ANN
Threshold exceeded?
No
Yes
-Add the current contingency to the list of critical cases
Run a Load Flow for every critical contingency
END

Fig. 1. Flow diagram of the hybrid security assessment approach.

MW (this state corresponds to a snapshot saved on April 27, 1994 during the peak load situation). Excluding radial and external branches, a total of 110 contingencies are of interest (in this study, only branch outages will be considered, but generator outages could be very easily included since they do not involve topology changes). By linearly decreasing the load demand, and by resorting to an optimal power flow to find reasonable generation patterns, 18 additional states were obtained. Each of the resulting states, combined with all possible branch outages, yield a total of 2109 cases $(19 \cdot 111)$ which were used to train the NN. After the training process, that took 42500 iterations, the training set was aplied again to the NN resulting in 3 false alarmas (0.14%) and no missed detections. This means that only 3 extra cases would have to be numerically analyzed in addition to the 327 contingencies which are truly critical. Note that all cases belonging to the training set referred to contingencies affecting the same base-case topology.

The test set was generated intentionally from 8 different topologies for which the NN had not been previously trained. These alternative topologies were obtained from the base case by disconnecting a single line or transformer. In addition, 4 different loading levels were considered for every topology. Combining the 32 resulting states with all possible contingencies, and discarding a few infeasible situations, a total of 3472 test cases were applied to the NN, of which 725 corresponded to critical contingencies. The MLP response included a single missed detection and 146 false alarms (4.2%). The missed detection turned out to be irrelevant in practice, because the only overload detected referred to a line carrying 190.8 MW. whose rating was 189 MW. The relatively high proportion of false alarms suggests that a more accurate way of modeling topological changes should be devised. However, thanks to the use of an ANN that was just trained with the base-case topology, the load flow program had to be run only 871 times, out of the 3472 tested cases, which means a saving of 75% in the computational effort.

4 Conclusions

In this paper, the potential application of a MLP to the screening phase of security assessment is explored in the context of realistically sized power transmission systems. Firstly, an ANN based on a selected subset of branch power flows is developed, which is capable of simultaneously dealing with a reduced number of major contingencies. Although the performance obtained with this approach is acceptable, its generalization to all possible contingencies and network topologies is not trivial. Secondly, an alternative scheme is proposed based on bus power injections, which must be properly modified to account for topological modifications and/or outaged branches. This generalized approach has shown to perform satisfactorily so long as the input topology is close enough to the base-case topology used in the training phase, which is usually the case. Further research efforts should be directed to devise more accurate ways of compensating power injections for topological changes.

References

1. R. Bacher and W.F. Tinney, "Faster Local Power Solutions: The Zero Mismatch Approach", IEEE Trans. Power Syst., Vol. PWRS-4, pp. 1345-1354, Nov. 1989.
2. N. Balu, T. Bertram, A. Bose, V. Brandwajn, G. Cauley, D. Curtice, A. Fouad, L. Fink, M.G. Lauby, B.F. Wollemberg and J.N. Wrubel, "On-Line Power System Security Analysis", Proceedings of the IEEE, Vol. 80, no. 2, February 1992, pp. 262–280.
3. V. Brandwajn, "Efficient Bounding Method for Linear Contingency Analysis", IEEE Trans. Power Syst., Vol. PWRS-2, pp. 38-43, Feb. 1988.
4. V. Brandwajn and M.G. Lauby, "Complete Bounding for AC Contingency Analysis", IEEE Trans. Power Syst., Vol. PWRS-4, pp. 724-729, May 1990.
5. CIGRE Task Force 38-06-06 on Artificial Neural Network Applications for Power System. Convenor Dagmar Niebur, Vol. 1 No. 3, pp. 133-157, Dec. 1993.
6. G.C. Ejebe and B.F. Wollemberg, "Automatic Contingency Selection", IEEE Trans. Power App. Syst., Vol. PAS-98, pp. 92-104, Jan/Feb. 1979.
7. Fischl et al., "Screening Power System contingencies Using Back-propagation Trained Multiperceptron", Proc. of the 1989 ISCAS, Vol. 3, pp. 486-489, 1989.
8. Mohamed-El-Sharkawi and D. Niebur, editors, "Tutorial Course on Artificial Neural Networks with Applications to Power Systems", IEEE Catalog Number: 96 Tp 112-0.
9. W.Y. Ng, "Generalized Generation Distribution Factors for Power System Security Evaluation", IEEE Trans. Power App. Syst., Vol. PAS-100, pp. 1001-1005, Mar. 1981.
10. D. Niebur and A. Germond, "Power System Static Security Assessment Using Kohonen Neural Network Classifier", IEEE Trans. Power Syst., Vol. 7 , pp. 865-872, May 1992.
11. A.P. Sakis Meliopoulos, C.S. Cheng and F. Xia, "Performance Evaluation of Static Security Analysis Methods", IEEE Trans. Power Syst., Vol. PWRS-9, pp. 1441-1449, Aug. 1994.
12. P.W. Sauer, "On the Formulation of Power Distribution Factors for Linear Load Flow Methods", IEEE Trans. Power App. Syst., Vol. PAS-100, pp. 764-770, Feb. 1981.
13. B. Stott, O. Alsac and A.J. Monticelli, "Security Analysis and Optimization", Proceedings of the IEEE, Vol. 75, no. 12, pp. 1623-1644, December 1987.
14. W. F. Tinney, "Compensation methods with with ordered triangular factorization", IEEE Trans. Power App. Syst., vol. PAS-91, pp. 123-127, Jan./Feb. 1972.
15. J. Zaborsky, F.W. Whang and Prasad, "Fast Contingency Evaluation Using Concentric Relaxation", IEEE Trans. Power App. Syst., Vol. PAS-99, pp. 28-36, Jan/Feb. 1980.

On Line Industrial Diagnosis: An Attempt to Apply Artificial Intelligence Techniques to Process Control

Alonso González, Carlos[1], Pulido Junquera, Belarmino[1], and Acosta, Gerardo[2]

[1] Dept. de Informática, Facultad de Ciencias, Universidad de Valladolid
Edificio de las Tecnologías de la Información y las Comunicaciones
Campus Miguel Delibes. Camino del Cementerio s/n.
E-47011 Valladolid. Spain
TE: +34 83 423670
FAX: +34 83 4233671
e-mail: calonso, belar@infor.uva.es

[2] Dept. de Electromecánica, Facultad de Ingeniería.
Universidad Nacional del Centro de la Provincia de Buenos Aires (UNCPBA)
Av. del Valle 5737 - (7400) Olavarría, Argentina.
TE: +54 284 51055/6
FAX: +54 284 50628
e-mail: ggacosta@fio.unicen.edu.ar

Abstract. Three Knowledge Based Systems (KBS's), performing diagnosis and integrated in a Knowledge Based Supervisory System (KBSS), are presented. The systems work on line in a continuos process factory and one of them is routinely used at the control room. The paper summarises the conceptual framework that guided the design of the KBSS, describing later the fault identification module of each diagnostician. Specially relevant were the different approaches tried to deal with the dynamic nature of the problem, looking for a good trade off between expressiveness and simplicity of the final system. Some experimental results, obtained from actual system performance at a beet sugar factory, and major conclusions, are included.

1 Introduction

This paper describes the evolution of the diagnosis module of a KBSS that supervises on line a continuous process: a beet sugar factory. Three diagnosis systems were developed: AEROLID [5, 6], TEKNOLID[2], and TURBOLID[11]. The systems differed, mainly, on the solution proposed to manage temporal information. But before describing them, we will introduce the conceptual framework in which they are embedded.

The factory to be supervised is almost fully automated (necessary condition) via a Distributed Control System (DCS) and is managed from a central control room by few operators. The KBSS was in charge of what we have called Global Supervision[3] to indicate that this supervision is not limited to the control loop scope. From the several tasks, [3], that could be contemplated at this level, we found that to support a diagnosis system four tasks should be necessarily performed: Monitoring, Operation Mode, Fault Diagnosis and Man Machine Interface (MMI). Monitoring takes care of fault detection. Operation Mode checks if the working environment of a device is correct, in order to allow it to obtain its expected functionality (controller tunings, proper set points...). Fault Diagnosis task tries to identify the anomalous behaviour of process units due to physical faults, finding the faulty components; this task is usually called fault diagnosis in the literature [9, 13]. Finally, MMI decides on what circumstances the control room operator should be informed of the detected problems.

The developed KBSS had a blackboard architecture with several modules supporting various tasks: data-reconciliation module, predictive control module... But only three modules were related to diagnosis tasks: Monitoring, Fault Identification and Alarm Management Modules. Monitoring was common for the three diagnosticians developed. Fault Identification carried out Operation Mode and Fault Diagnosis tasks, been different for each diagnostician. The systems also differed on the Alarm Management Module, but this aspect will not be discussed in this paper.

2 The Process Under Supervision

The purpose of the process is to produce sugar from beet. It consists basically of four main sections, Diffusion, Purification, Evaporation, and Crystallisation. Briefly speaking, raw pieces of beet go into a cylindrical machine, the Diffuser, by one end, while by the other side, water (@ 70 °C) is introduced to obtain a juice rich on saccharose (15%) and impurities (non-sugars). The latter are eliminated in Purification, in order to obtain a juice of great purity (95%). The next stage, Evaporation, see figure 1, increases the juice concentration extracting water by boiling, and producing syrup (65% of saccharose). Concentration is further increased in batch units (crystallisators) until high enough for sugar crystals to grow. Though we can not give further details about the criteria to drive the process, the point is that once an operation policy is adopted, the factory is expected to keep on working in a nominal range. From this arises the importance of diagnosis to detect and explain abnormal process behaviour, and when possible, propose a repairing protocol.

3 Development and Run-Time Environments

The KBS's described here were implemented in G2(Gensym). The knowledge representation language supports rules, frames, methods and a Pascal like procedural language, extended with some real time oriented facilities. The system was developed and tested at the faculty laboratory, where a dynamic, non linear simulation of the plant, with models obtained from actual factory data, had been developed [1]. The development environment consisted of two Sparc WS running Solaris (UNIX), communicating each other through an Ethernet. One supported

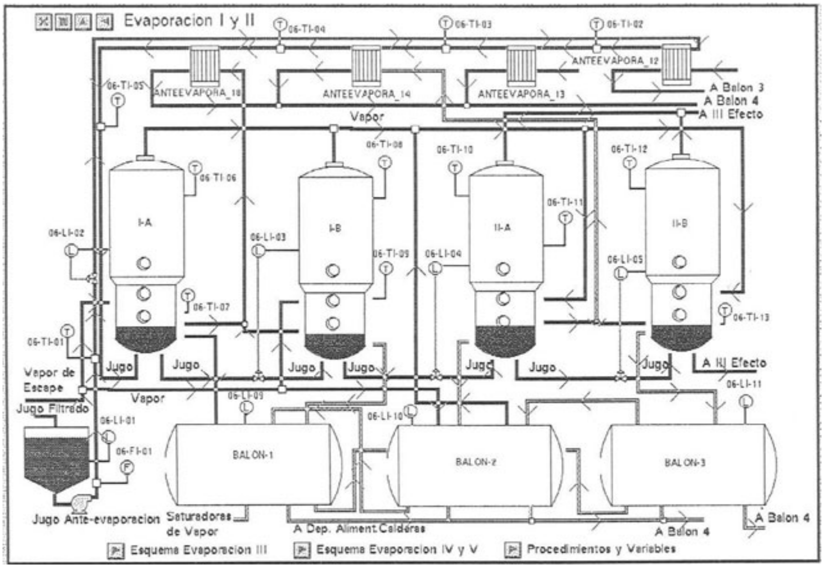


Fig. 1. One scheme of the evaporation section as would be seen using MMI

the KBS, in the same conditions as in factory, that is, reading and writing data from the net. The other played the part of the process in real time. As regards as the run-time environment, the KBS run in a HP WS, again under UNIX, at the control room. Now the system collects data from a real time database, which is a kind of buffer between the DCS bus and the KBS.

Another important feature of the tool was the man-machine interface. The developed KBS's exhibit mimics like a SCADA system, shown in Figure 1, from which the user could have a feeling of what was going on at the plant.

4 Monitoring

Fault detection is a crucial step in any diagnosis system working with dynamic processes [10, 8]. These systems resorted to the concept of Monitored Variable, MV, [5, 6], to detect anomalous situations. MV's are symptoms with great discrimination power and well known nominal ranges. Typical examples are *"level of diffuser's vase oscillating"* or *"level of diffuser's vase with increasing trend"*. To state potential problems, MV's were checked against fixed thresholds. As a consequence the system could only supervise the process when working on nominal conditions. To cope with transitories and also to further assert the detection, a three threshold scheme was used: vigilance, triggering and recovering. Crossing of vigilance threshold invoked a new task named Intensive Monitoring. This task demanded a persistence condition over vigilance and may include specific knowledge to confirm the detection. Only if the triggering threshold was crossed and the previous conditions were satisfied, the state of the MV become critical and the detection was confirmed. We found that this scheme provided a very robust detection method. Also remarkable was the fact that the set of MV allowed a very focused diagnosis in a rather complex system. The major problem we found was the interrelation among MV, not modelled at this level, that had to be solved by the diagnosis module.

5 First Solution: AEROLID

AEROLID, [5, 6], was the first diagnosis system developed. Hence, it was important to achieve two objectives: to allow an easy transfer of technology to the factory technicians and to demonstrate the utility of the tool. This explains the scope of the system: limited to the diffusion section, classic expert system approach, no explicit temporal model and singled fault hypothesis per MV.

5.1 Fault Identification

AEROLID did not discern between Operation Mode and Fault Diagnosis: the system searched the origin of detected problems with the same technique, independently of the nature of its cause.

The diagnosis task employed cause-effects trees that modelled the casual relationships among symptoms (effects) and causes. Each tree was implicitly defined by a set of rules with syntax:

> **if** $\{$*symptom* **and**$\}^+$ *symptom* **then** *cause*$_1$ **end if** *cause*$_1$ **and** $\{$*symptom* **and**$\}^+$ *symptom* **then** *cause*$_2$ **end**

The system backward chained these rules to find the original causes (leaves of the trees) from group of symptoms, selecting the first cause reached, according to the single fault hypothesis.

Notice that though the underlying causal model is represented by wff. 1, the rules mirror wff. 2

$$C \supset S_1 \wedge S_2 \wedge \ldots S_i \ (wff.\ 1)$$

$$S_1 \vee S_2 \vee \ldots S_i \supset C \ (wff.\ 2)$$

The whole diagnosis knowledge was hierarchically organised: factory, section and MV level. Then, each MV was associated with the cause-effect sub-tree that included the causes that could explain the MV behaviour. Also, to take into account that the same fault might affect various MV's, cause-effect relations were allowed between leaves of different trees.

Regarding time, the system resorted to a point-based linear time model [4, 7]. Only present and historical data were analysed, and temporal relationships were managed adding to symptoms and causes a time tag.

Finally, no numerical technique [14] was used to cope with uncertainty. In fact, the system converted numerical data to precise symbols employing, at least, three tresholds, in a similar way as monitoring did it. Hence, imprecision to define a boundary was tackled defining an interval with an associated symbolic value. That way, for those numerical values affected by uncertainty, there was a symbol to denote that their values were contained in the intervals where no clear decision could be taken. Consider also that every numerical data was subjected to a modal consistency check (range of magnitude and first derivative) to prevent spurious data to enter the system.

5.2 The diagnosis process

Whenever the state of a MV changed to critical, a diagnosis process was launched and the set of rules associated with the MV was invoked. If a cause could be stated, it would be considered as the diagnosis. Otherwise, if the set of rules could not conclude any cause, the diagnosis would be "unknown" but, at least, the operator could be informed of the detected problem. This diagnosis object lived in the system while the MV persisted on critical state, and some time after MV reached normal state, because casual relationships among different MV's were supported by it. If later, a new MV became critical and there existed a diagnosis pointing to its sub-tree, it was the first cause that the system tried, in an effort to explain the present anomalies with the smallest number of diagnoses coherent with the current symptoms.

5.3 Experimental results

This systems was tested at plant during the '93-'94 campaign, presenting a questionnaire to the control room operators, to be filled after acknowledgement. The results, obtained from five shifts, are summarised in Figure 2.

COMPLETENESS (over a total of 97 diagnoses)	Known Cause	81.5 %
	Unknown Cause	18.5 %
CONSISTENCY (over a total of 45.4% acknowledged alarms)	Correct	79.6 %
	Incorrect	4.5 %
	Unknown	15.9 %

Fig. 2. Results obtained by AEROLID during the 1993-94 campaign

The low rate of acknowledgement might be due to the lacking of an elaborated MMI, that informed of every diagnosis found. The success rate among acknowledged alarms showed the viability of this approach, despite of the assumed simplification hypothesis, mentioned earlier.

6 Stronger Inference and Temporal Information: TEKNOLID

The main limitation of AEROLID was that it could only diagnose causes which effects had a similar or quicker dynamic than the associated MV. Another undesirable feature is the reliance on abduction, which may be an unsound inference rule. With TEKNOLID we tried to tackle these shortcomings.

6.1 Fault Identification

Applying contraposition to wff. 1, we obtain wff. 3, that allows to reject a cause when any of its symptoms is not present, using modus tollens, a sound inference rule, as the basic inference step:

$$\neg S_1 \wedge \neg S_2 \wedge \ldots \wedge \neg S_i \supset \neg C \ (wff.\ 3)$$

However, this expression is not yet fully descriptive because there is a clear allusion to time in causation. The problem was managed allowing causes as well as symptoms to have temporal ubiquity resorting to a point-based linear time model. Each cause-type had associated a set of symptoms (effect-type) with the time intervals, relative to the cause, where they must be observable. Similarly, each MV had a list of possible causes-type plus a time reference relative to the MV detection. Then, when a problem is detected, the KBS created a time map with cause-tokens and effect-tokens [7]. Diagnosis basically consisted of rejection of causes, if their symptoms were not observed when it was required by the time map.

6.2 The diagnosis process

As a MV changed its state from normal, the diagnosis process created a time map for each of its associated causes, considering them as possible candidates while they had not been rejected. A procedure, named filtering, periodically checked the status of the causes, rejecting them if any of its effect-tokens become false. The process finished if no cause persisted, unknown diagnosis, or when the maximum time to be observed, defined for each MV, had elapsed. Meanwhile, the surviving causes were ordered according to a fuzzy criterion, obtained from the membership grade of the observed effects. Hence, the diagnosis process behaved as an "any time algorithm": from the beginning of the detection, it had available an ordered list of still possible cause, focussing the diagnosis with time. This list was presented to the operators and dynamically updated. Operators could also reject causes, being naturally integrated in the process. Also remarkable was that this strategy naturally allowed diagnosing multiple faults, as long as their symptoms did not overlap.

6.3 Experimental results

This system was successfully tested at plant, January 95 and identified complex faults, like "connection vapour-steam through a crystallisator" or even "partially open bypass at the juice input of evaporator IV", not on the scope of AEROLID due to its dynamics. However, we could not elaborate statistics of plant behaviour. The reason was that, surprisingly, plant technicians considered that performance improvement did not justify neither the temporal information elicitation effort nor the increased complexity of the diagnostician.

7 A Pragmatic Balance: TURBOLID

TURBOLID, the last developed system, was designed looking for a practical balance between the simple design of AEROLID and the more sophisticated capabilities of TEKNOLID. Factory managers wanted an AEROLID like tool, but with the capability of analysing problems with slower dynamics than those manifested by the MV's. Hence TURBOLID was conceived as an extension of TURBOLID. It used single fault hypothesis and a similar diagnosis technique, but it was extended to accommodate the new demands and improved via a diagnosis protocol.

7.1 Fault Identification

TURBOLID explicitly separates Operation Mode from Fault Diagnosis. Operation Mode task could be considered as the task that tested for working conditions required by the diagnosis. It seemed naturally to check these conditions before diagnosis proceeded. Look-up tables, that contained the required settings, plus procedural code that was invoked to verify the current settings, supported this kind of testing. Fault diagnosis was tackled in a similar way as AEROLID did, but refined to easily accommodate references to future time. In TURBOLID, every MV had also associated a cause-effect sub-tree, with references to past and present data. These symptoms, that had to be observable when the state of a MV become critical, were generically called fast symptoms. To allow the detection of causes with symptoms that manifested themselves after the MV had become critical, slow causes, TURBOLID expanded the cause-effect sub-trees representation, identifying possible causes: set of causes with common fast symptoms already present. Therefore, in a first diagnosis step, TURBOLID could conclude a single cause or a list of possible causes. The discrimination between these possible causes was done in a second step, which waited a certain amount of time for the symptoms with slower dynamic than the associated MV, slow symptoms, to be detected (or rejected). These information was, again, organised as a tree, one for each possible causes of the list, with only one level, and represented by rules of the form:

> **if there exists** *a possible cause* **and** $\{$*slow symptom* **and**$\}^{+}$ *slow symptom*
> **then** *slow cause* **end**

As the slow symptoms included a time tag, this scheme allows for a pattern of ordered symptoms to be searched for. The temporal simplifying assumption used by TURBOLID was: the time distance of the pattern of slow symptoms respect to the VM detection is irrelevant, as long as it can be found before the time to be present (defined for each slow cause).

7.2 The diagnosis process

Again, diagnosis started when the state of a VM became critical. But TURBOLID employed an explicit diagnosis protocol that comprised four phases, sequentially organised. Resorting to the single fault hypothesis, as soon as any of them found a diagnostic, the process finished. The phases of the protocol were:

- *Initial Phase*: which was restricted to consider only Operation Mode.
- *Instant Phase*: which looked for diagnosis explainable only with fast symptoms and, eventually, creating the list of possible causes.
- *Slow Phase*: diagnosis related to slow symptoms for which possible causes were already stated.
- *Discard Phase*: which looked for a compromise between announcing an unknown cause or giving clues to the operator using information that could not be considered totally conclusive.

The Slow Phase deserves some additional comments. The rules included on it were periodically invoked by the system, not backward chained. This direct invocation was maintained until a slow cause was asserted or the list of possible causes was empty. Any possible causes was extracted from the list when the maximum of the time to be present of its associated causes had elapsed.

7.3 Experimental results

This system was tested during the last month of the 1995-96 campaign. The results obtained from five shifts, as judged by the operators, are summarised in Figure 3 which shows that TURBOLID obtained a similar diagnosis success rate as AER0LID: 81,5% versus 79.6% respectively. But if we consider that AEROLID only worked on the diffusion section, while TURBOLID included also purification and evaporation, and we compare the results by sections, we observe that: on diffusion and purification, TURBOLID reached an 87.5 % of success; on evaporation section, it reached a 78.9 % of success rate, similar to the rate obtained by AEROLID on the diffusion section.

This was considered a satisfactory result because on the evaporation section numerous faults showed a complex dynamic, which could not be properly diagnosed by AEROLID.

8 Conclusions

The most remarkable result is that the last diagnosis system, TURBOLID, is a routine tool at the control room. Perhaps the main contribution of this work

	Fault Detection			Diagnoses			
	Total	Correct	Incorrect	Total	Correct	Incorrect	Unknown
Diffusion & Purification	10	8	2	8	7	1	0
Evaporation	22	19	3	19	15	2	2
Total	32	27	5	27	22	3	2
%	100	84,37	15,62	100	81,48	11,1	7,4

Fig. 3. Results achieved by TURBOLID in the 1995-96 campaign

may be to analyse the reasons of its success. The rate of correct diagnosis, approximately 80%, is one of the motives. But not the only one, because we believe that TEKNOLID could have improved it. In our opinion, and considering only aspects related to the diagnosis task, the other main reason is its simple design principles. TURBOLID is a system very easy to develop and maintain and, however, allows capturing enough knowledge to diagnose on line a continuous process of middle complexity. The design principles that, we think, explain its success could be summarised as:

- The use of MV conceptually divides the system, in such a way that the diagnosis of a very complex system is replaced by numerous diagnosis of simple subsystems.
- The simple management of temporal information that suggests that not always a better temporal representation will produce the better solution. In this approach, it seems that we found a good balance between expressiveness and complexity. We do not claim that it may be extended to any continuos process, but we think that this solution is easily transferable to many of them.
- Finally, the presence of the diagnosis protocol, that resembles the way operators diagnose problems, separating Operation Mode from Fault Diagnosis and including a phase that examines the plant evolution.

Of course, the three discussed systems are limited: strong device dependence, single fault hypothesis, only works on nominal conditions ... This problems are also consequence of its own design principles. We firmly believe that to overcome these limitations, another diagnosis methodology is needed. Currently, our diagnosis group [12] is working on Consistency-Based Diagnosis [8] trying to integrate it in an architecture with a KBS running in parallel with the diagnosis protocol and interacting with it. But we are afraid that a great effort has still to be done before this methodology may produce a tool comparable to TURBOLID.

Acknowledgements This work was financially supported by SGAE., SA. We want to specially acknowledge J. Barahona, A. Bastos, M. Bollain, L. Llorente and R. Vargas. Our research group wishes to acknowledge the support during the last months from the CYTED project "Técnicas de Inteligencia Artificial para Supervisión, Diagnosis y Control de Procesos".

References

1. Acebes, L. F., M^{a}. T. Alvarez, J. Achirica, C. Alonso, G. Acosta, C. de Prada: A Simulator to Validate Fault Detection in an Industrial Process with Expert System. Proc.de la Int'l Conference on Simulation of Continuous Processes, Barcelona, Spain, June 1-3. (1994) 709–713.
2. Acosta, G., Alonso, C., Acebes, L.F., Sánchez, A, and de Prada, C.: Knowledge based diagnosis: dealing with fault modes and temporal constraints. In Proc. de IEEE/IECON, Boloña, Italia, Vol.2, (1994) 1419–1424.
3. Acosta, G.G. y Alonso González, C.: Towards a task taxonomy in Knowledge Based Systems for the Process Control Supervision. In Proc. of the II Congreso Internacional de Informática y Telecomunicaciones, IV Simposio de Inteligencia Artificial (INFOCOM96), Bs. As., Argentina, Junio 10-14, (1996) 316–325.
4. Allen, J.: Maintaning knowledge about temporal intervals. Communications of the ACM, vol. 26, **11**. (1983) 832–843.
5. Alonso, C., Acosta, G. , de Prada, C., and Mira Mira, J.: "A Knowledge Based Approach to Fault Detection and Diagnosis in Industrial Processes: A Case Study". In Proc. of the IEEE ISIE'94-Santiago, Chile May 25-27. (1994) 397–402
6. Alonso, C., Acosta, G., Mira, J., and de Prada, C.: Knowledge based process control supervision and Diagnosis: The AEROLID approach. To appear in Expert Systems with Applications, vol. **8**. (1998)
7. Charniak, E., and McDermott, D.: Introduction to Artificial Intelligence: Addison-Wesley. (1985)
8. Dressler, O. and Struss, P.: The consistency-based approach to automated diagnosis of devices, as a chapter in Principles of knowledge representation. Gerhard Brewka ed. (1996) 269–314.
9. Patton, R., Frank, P., and Clark, R. (Eds.): Fault diagnosis in dynamic systems. Theory and applications. Prentice Hall International. (1989)
10. Pouliezos, A.D, and Stavrakakis, G.S.: Real Time Fault Monitoring of Industrial Processes. Kluwer Academic Publishers. (1994)
11. Pulido, J.B., Acosta, G., Llamas, C., and Alonso, C.: TURBOLID: A KBS considering temporal information and performing diagnosis in a beet sugar factory. In the Proc. of the Intl. Conf. of Engineering of Intelligent Systems, EIS98, Feb. 10-13, Tenerife, Spain. (1998)
12. Pulido, J.B., Acebes, F., and Alonso, C.: Exploiting knowledge about Structure and Behaviour in Consistency-based diagnosis with Fault Modes in Dynamic Systems. In the Proc. of the Intl. Conf. of Engineering of Intelligent Systems, EIS98, Feb. 10-13, Tenerife, Spain. (1998)
13. Rehbein, D., Thorp, S., Deitz, D. and Schulz, L. Expert Systems in Process Control. ISA Transactions, Vol. 31, **2**. (1992) 44–49.
14. Smets, P., Mamdani, E.H., Dubois, D., and Prade, H (Eds.): Non-Standard Logics for Automated Reasoning. Academic Press. (1988)

An AI Modelling Approach to Understanding a Complex Manufacturing Process

M.A. Rodrigues and L. Bottaci
Artificial Intelligence and Pattern Recognition Research Group
Department of Computer Science
The University of Hull
Hull HU6 7RX, UK
http://www.enc.hull.ac.uk/CS/aise/aise.html

Abstract

A manufacturer who wishes to set an improved quality target has the problem of setting the target high enough to make a significant improvement in customer satisfaction but at the same time, not so high that the cost of rejected products is counter productive. The problem is difficult because of the uncertain relationship between the quality level, customer satisfaction and cost. This paper describes an approach to understanding the complex relationship between quality target, customer satisfaction and cost for a manually intensive production line. An AI-based modelling and simulation approach is used to investigate in a quantitative way the consequences of changing manufacturing parameters. The approach allows the production engineers to improve their understanding of the production process by formalising their knowledge in a model that can be tested in a variety of scenarios.

Keywords: Manufacturing, Monitoring, Fault Diagnosis, Simulation, Validation.

1 Introduction

This paper is concerned with understanding and improving a manufacturing process. Our industrial partner is a market leader in the manufacturing of caravans which are built in small batches on a manually intensive production line. We are particularly concerned with how best to increase overall customer satisfaction and what are the important manufacturing parameters that influence the level of internal rejections. Although one would expect a higher level of quality to lead to increased customer satisfaction, the relationship between these two factors is not straightforward. In an attempt to understand this relationship, warranty information and surveys of customer satisfaction are commonly used.

The desire to increase quality must be balanced by the desire to control cost. In this paper we consider a simple quality mechanism in which the quality of the product is effectively controlled by a test and inspection process. Products that do not achieve the required standard are rejected and subjected to additional work. Clearly, a higher reject rate will incur a higher cost. Howerver, the relationship between the set quality target and actual reject rate is not a clear cut because, over time, even without intervening in the production process, workers are able to adapt to new quality targets.

The overall questions that we are asking are, how should the test and inspection targets be set to increase overall customer satisfaction and what is the overall cost in terms of both product rejection (target quality level high), and loss of reputation and future business (target quality level low)? Related questions include: how might these effects be measured; how can we make effective use of quantifiable information such as accumulated test data and warranty claims; what kind of feedback is required from the customer? In general, we seek understanding of a process involving flow of materials, manual assembly operations, equipment testing procedures and inspection, in addition to labour force and customer's perception of quality. The complexity of the problem is increased by the inclusion of human factors in the process loop. It is necessary to include these because the production is manually intensive.

We are thus faced with a complex modelling problem involving several interdependent variables. A number of Artificial Intelligence (AI) techniques lend themselves to this kind of problem (e.g. [3], [6], [9]). Such techniques include constraint satisfaction, means-ends analysis, goal-directed behaviour and semantic nets, in which the meaning of a concept is represented in its relationships with other concepts. Attempts at cognitive modelling in manufacturing environments have provided a limited degree of understanding of the processes involved regarding human interactions (e.g. [5], [7], [14]). In the context of monitoring, control, and diagnosis of manufacturing defects expert systems technology has been largely used (e.g. [1], [2], [4], [8], [15]). We chose to adopt an AI approach to this problem by modelling the process as a number of interacting entities. Entities are dynamic and goal-directed but operate within given constraints. This has implications both to the way modelling is carried out and how an entity's output is evaluated. We describe a case study concerning electrical appliances but the method is general and intended to be useful for more complex processes. We intend to test it by application to the production of a complete caravan. Current test procedures, our method and the modelling tool are described in Section 2, the model and results are described in Section 3, and a discussion is presented in Section 4.

2 The Method

We are using the testing of electrical appliances in a caravan as a case study to illustrate our approach. The present testing process is largely qualitative rather than quantitative. For example, the lights are turned on and off, the refrigerator is switch on for a short period to check that it becomes cold. Thus, the tests show only that the appliances are working, they do not show how well they are working. It is proposed to closely monitor a number of electrical tests as there are 22 appliances in a typical caravan and power consumption varies for each appliance from 58mA for the television aerial amplifier to 8A for the heater as illustrated in Table 1. Testing will involve measuring and recording current flows through specific circuits and comparing these with given tolerance thresholds.

Quality targets in this situation can be expressed in terms of tolerance values. Initially, tolerance values must be estimated. Once in operation, however, feedback provided by customers may be used to adjust these values. Unfortunately, customer feedback is not simple to acquire or use. The following are some of the issues that need to be considered:

- Different customers have different perceptions of quality;
- How will overall customer satisfaction change in response to changes to various test parameters?
- Is there a danger that if test parameters are changed too drastically the production process will become unstable?
- What is the cost of producing a given improvement in quality?

We have designed and implemented an AI-based visual programming environment for dynamic systems modelling [10], [11] named Prox. Prox is implemented in Pop11 (an AI language similar to Lisp) and also inherits all the Pop11 language constructs and functionality. Prox facilitates a rigorous analysis of dynamic processes both with and without prior knowledge of the integro-differential equations that describe the behaviour of the system under study. It allows the modeller

Condition	Current Amps a.c.
All systems off, source switched to mains	0.055
Battery charger (up to)	5.0
Lumo lights - forward	0.1
Lumo lights - forward and aft	0.14
Toilet light	0.09
Spotlights - one on	0.11
Spotlights - two on	0.16
Kitchen light	0.088
Awning light	0.14
Water pump	0.16
Refrigerator	0.52
Aerial booster	0.058
Omnivent setting 1	0.14
Omnivent setting 2	0.17
Omnivent setting 3	0.35
Heater fan	0.16
Heater slow	4.0
Heater fast	8.0
Water heater	2.9
Cooker igniter	0.059

Table 1: Electrical test parameters: current consumption in Amperes on a typical caravan.

to acquire an intuition for the mechanisms that give rise to a particular behaviour pattern. It helps to explain how such processes work and what are the contributing factors and their interrelationships.

Our modelling method is systematic and consists of building a semantic net like structure in which some entities are goal-directed with given constraints. It consists of the following steps:

1. Identify the variables or entities that need to be controlled (i.e. the entities that are goal-directed).

2. Set a goal, reference or target value for each controlled variable.

3. Define functions to read the actual value of the controlled variables (usually from the environment). From the point of view of the system, these functions are input functions.

4. Define functions to set parameters that have an effect, (usually indirectly through the environment) on the value of the controlled variables. From the point of view of the control system, these functions are output functions.

5. Adjust the control parameters to obtain the required characteristics of the control system (e.g. gain and error sensitivity).

The AI environment in which Prox has been implemented (Poplog) renders the method more flexible and open to experimentation as it can be seamlessly interfaced with other AI languages such as Lisp and Prolog. Moreover, as soon as a model is built (by describing the structure of the system using a visual interface and setting initial values for some of the functions) it can be immediately run. The outputs of the model describe how variables change as function of time.

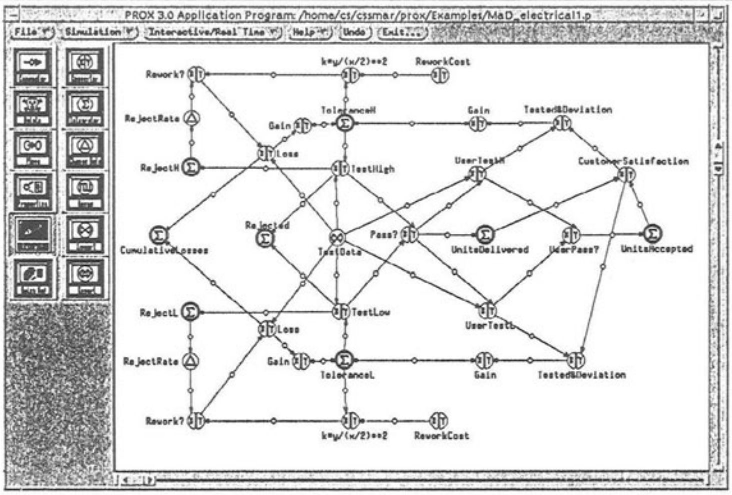


Figure 1: The Prox model for electrical appliance testing.

3 Results

Figure 1 depicts a model built to understand the relationship between quality target, customer satisfaction and cost in the context of electrical appliances. Test data and threshold parameters for a refrigerator were used. In order to understand the model, we start at the central node named "TestData" and look at the right side of the model. "TestData" is a file containing values of the electrical current in Amperes, read by a testing equipment. In this model, each current value represents an appliance. This value must be within a tolerance band, the high limit and the low limit. The high limit is tested in the "TestHigh" which contains a logic relationship whose output is 1 or 0. The "Pass?" node accumulates a one for each appliance that passes both tests ("TestHigh" and "TestLow"). The integrator "UnitsDelivered" counts the number of passes.[1]

The model assumes that the customer's perception of quality is more demanding than that of the manufacturer. An appliance typically has a higher probability of malfunctioning when it is with the user than when it is with the manufacturer. A customer will therefore either accept the appliance, in which case it is added to the integrator "UnitsAccepted" or reject it with additional information giving an indication of how far from the customer reference the results of the tests are, "Tested&Deviation". This is then used to adjust the tolerance bands "ToleranceH" and "ToleranceL" which are adjusted as follows:

$$Tolerance_{(t+dt)} = Tolerance_t - G \tag{1}$$

where $G = G_1 + G_2$ is the overall gain, G_1 is the gain due to customer satisfaction and G_2 is the gain due to internal losses (defined by Equation 4). G_1 is defined as:

$$G_1 = D_i/1000, \tag{2}$$

where i is the current simulation cycle and D is the deviation of "CustomerSatisfaction" from its ideal value. The reference value for the deviation D is zero, so that if D is positive, Equation 1 will drive the value of tolerance down; otherwise it will increase.

[1] This node is actually an Euler integrator with the following general equation: $UnitsDelivered_{(t+dt)} = UnitsDelivered_{(t)} + Pass * dt$.

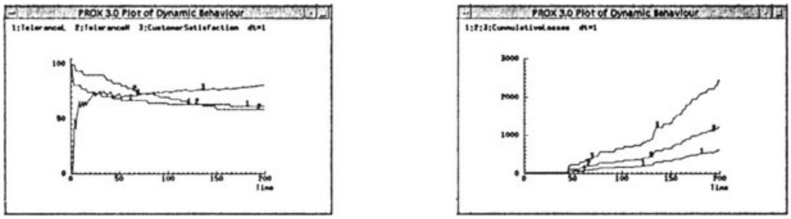


Figure 2: The curves show relationships between customer satisfaction, test parameters, and internal losses. Left, customer satisfaction only is used to drive the value of the tolerances down. The consequence is a high number of internal rejections depicted as losses on the right.

Since the low and high thresholds are independent variables, it is desirable that any customer who rejects an appliance also provide some additional information regarding the direction in which to adjust the high or low threshold. For instance, if a refrigerator is said not to be cooling fast enough, this has to be described as "increase/decrease current". Blind variation of parameters may mean slow or no convergence.

The left side of the model deals with the cost of internal rejections. If a test is unduly strict, there will be a high proportion of rejects with associated costs. On the other hand, if a test is slack, customer satisfaction might decrease and this also has associated costs. There is always a trade-off between customer satisfaction and internal rejection, described as the cost of defect prevention or quality improvement. In this respect, our approach is similar to the approach described in [12] and [13], in which the overall loss, L, caused by deviation of a single variable characteristic v from the nominal value is:

$$L(v) = Kv^2 \tag{3}$$

where the constant $K = C/T^2$, where C is the cost of scrapping or reworking an individual product, and T is half the tolerance band for the characteristic. These parameters are described in the model and represent a situation of 100% testing. Such a loss estimate has worked well in the model by contributing to the adjustment of testing parameters and to the stability of the model's behaviour.

Let us now consider how internal rejection costs influence the actual tolerance band. The gain described in Equation 1 depends on G_1 and G_2. At each control cycle the gain due to internal losses is adjusted as folows:

$$G_2 = (\mathit{Loss}/500) * \mathit{Tolerance}, \tag{4}$$

where ***Loss*** is the cost of rework for an item calculated according to Equation 3. The reference value for ***Loss*** is zero, so that if ***Loss*** is positive, it will drive the value of the tolerance up; if negative, the tolerance will decrease. Note that both the cost of rework (losses) and customer satisfaction are used to control the same tolerance (through adjusting twice the value of the gain G on every cycle – Equations 2 and 4). Eventually, a compromise or trend will be achieved for the tolerance bands.

If the model takes account only of customer satisfaction, the tolerance band is driven constantly to smaller and smaller values. Any positive error in customer satisfaction will have a positive (although small) drive applied right up to the point where the tolerance goes to zero. A situation to be welcomed by customers but disastrous for the manufacturer in terms of cost. Allowing the cost of internal rejections to influence the tolerance band produces a more realistic model. It also has the effect of damping tolerance correction so that in the long run the model and the manufacturing environment it describes are more stable. This is achieved by calculating the derivative of rejects given by:

$$RejectRate = \frac{RejectH_{(t)} - RejectH_{(t-dt)}}{dt} \tag{5}$$

which is then used to calculate the deviation v in Equation 3.

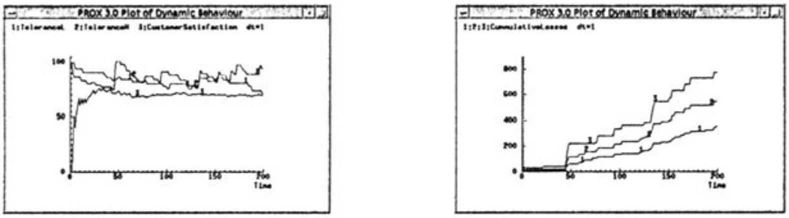


Figure 3: Changing model parameters. Left, tolerances have been adjusted to keep both rework costs down and customer satisfaction high. The consequence is a lower level of internal rejections depicted as losses on the right.

<table>
<tr><th colspan="4">Test data samples: 200, Initial tolerance: 10%</th></tr>
<tr><td>Rework Cost</td><td>Rework Cost considered?</td><td>Customer Satisfaction</td><td>Cummulative Losses</td></tr>
<tr><td>10</td><td>Yes</td><td>74.5</td><td>307.80</td></tr>
<tr><td>10</td><td>No</td><td>79.2</td><td>466.74</td></tr>
<tr><td>20</td><td>Yes</td><td>72.0</td><td>476.26</td></tr>
<tr><td>20</td><td>No</td><td>79.2</td><td>933.49</td></tr>
<tr><td>40</td><td>Yes</td><td>69.6</td><td>646.27</td></tr>
<tr><td>40</td><td>No</td><td>79.2</td><td>1866.98</td></tr>
<tr><th colspan="4">Test data samples: 200, Initial tolerance: 7.5%</th></tr>
<tr><td>Rework Cost</td><td>Rework Cost considered?</td><td>Customer Satisfaction</td><td>Cummulative Losses</td></tr>
<tr><td>10</td><td>Yes</td><td>78.2</td><td>390.89</td></tr>
<tr><td>10</td><td>No</td><td>85.7</td><td>588.31</td></tr>
<tr><td>20</td><td>Yes</td><td>74.5</td><td>607.53</td></tr>
<tr><td>20</td><td>No</td><td>85.7</td><td>1176.63</td></tr>
<tr><td>40</td><td>Yes</td><td>70.7</td><td>842.41</td></tr>
<tr><td>40</td><td>No</td><td>85.7</td><td>2353.25</td></tr>
<tr><th colspan="4">Test data samples: 200, Initial tolerance: 5%</th></tr>
<tr><td>Rework Cost</td><td>Rework Cost considered?</td><td>Customer Satisfaction</td><td>Cummulative Losses</td></tr>
<tr><td>10</td><td>Yes</td><td>82.0</td><td>579.01</td></tr>
<tr><td>10</td><td>No</td><td>100.0</td><td>1609.71</td></tr>
<tr><td>20</td><td>Yes</td><td>76.3</td><td>801.15</td></tr>
<tr><td>20</td><td>No</td><td>100.0</td><td>3219.42</td></tr>
<tr><td>40</td><td>Yes</td><td>71.6</td><td>1150.81</td></tr>
<tr><td>40</td><td>No</td><td>100.0</td><td>6438.85</td></tr>
</table>

Table 2: The effects of changing tolerance thresholds on customer satisfaction and internal costs.

Figure 2 depicts the general trend between customer satisfaction, tolerance bands and internal losses or rejections. The model has been simulated with a sample of 200 test measurements. On the left, note that there is an upward trend in customer satisfaction. This is because rework costs are not considered in this run of the model so that tolerances are driven to smaller and smaller values. The implications of such policy are dramatic for the manufacturer in terms of internal losses as shown on the right. The depicted curves for cummulative losses are for an initial tolerance of 10%, and rework cost at 10 (low cost, curve 1), 20 (medium, curve 2) and 40 (high, curve 3). The situation depicted is untainable by the manufacturer, as it generates a high and unrealistic number of internal rejections.

The model parameters were modified in a number of different ways such as depicted in Figure 3. In this run of the model, both customer satisfaction and level of internal rejections were used to control the same tolerance by adjusting the gains G_1 and G_2 in Equation 1. Although customer satisfaction has worsened in this case, this is a more realistic situation for the manufacturer, as internal rejections are within what is considered acceptable limits.

Analysis of the results from a number of alternative scenarios has show that if a relatively large tolerance of 10% is considered, then there will be the risk of a high number of rejects by the customer. This is reflected by an overall customer satisfaction of 79.2% as depicted in Table 2. Decreasing the tolerance band to 7.5% will increase customer satisfaction to 85.7% with a moderate increase in cumulative losses. However, if the tolerance band is tightened to 5%, then there might be no rejects from the customer i.e., 100% satisfaction, but at the price of a high level of internal rejects. The manufacturer has to take this into account for a particular product or appliance and may decide to risk releasing products that, although they have passed inspection tests, they may still face rejection by the customer.

4 Discussion and Further Work

This paper has described an AI-based approach to the modelling of quality control strategies. Our model is based on goal-directed behavioural entities whose inputs and outputs are interconnected. The model is executable and the resulting simulation is able to provide production engineers with an understanding of the factors that influence a complex production process. Each goal-directed entity may employ a range of behavioural strategies, such as constraint satisfaction, means-ends analysis and so on, as appropriate to the problem.

The model built to understand the consequences of adjusting test parameters of electrical appliances has demonstrated its usefulness to production and quality control processes. Production issues investigated include:

- Quantitative information such as costs can be accurately evaluated and the impact of changing various parameters can be assessed.
- The need to acquire from customer questionnaires, qualitative information that can be used to establish the direction in which test parameters should change. Most customers are in a position to provide such an indication.
- Difficult to assess costs such as loss of reputation and future business can be made an explicit part of the model.
- The relationship between production time scale, batch size and time lag in quality test data and customer feedback can be investigated. Certain control strategies require timely customer feedback which is expensive but may nonetheless be cost-effective.

The case study has also highlighted the generality of the technique:

- the existing simple model can be easily extended to include both more parameters and also to improve the sophistication with which existing parameters are modelled;
- the model is a useful tool for knowledge elicitation, as it allows the production engineer to formalise his or her qualitative knowledge in an executable model which can show both short and long term consequences of specific assumptions.

Our approach has advanced our understanding of the quality improvement problem and the results have demonstrated that the method can be applied to modelling other complex customer-production environment interactions. In particular, we intend to explore the effects of customer feedback delay; i.e. without changing any other model parameter, how such delays affect manufacturing parameters and, by extension, company strategy.

Acknowledgements

The work described in this paper is funded by the UK Engineering and Physical Sciences Research Council through grant EPSRC GR/L19508 in collaboration with the Swift Group.

References

[1] T. S. Arslan, L. Bottaci and G. E. Taylor, *A Fault Dictionary Based Expert System for Failure Diagnosis in a Multiple-PCB Environment*, Engineering Applications of Artificial Intelligence, Vol 6, no 5, pp 447-456, 1993.

[2] H.A. ElMaraghy and P.H. Gu, *Expert System for Inspection Planning*, Ann. CIRP, Vol 36, No 1 (1987) pp 85–89.

[3] G.W. Ernst and A. Newell, *GPS: A Case Study in Generality and Problem Solving*, New York:Academic Press, 1969.

[4] M. Ghallab and H. Philippe, *A Compiler for Real-Time Knowledge-Based Systems*, Proc. IEEE Int. Symposium on AI for Industrial Applications, pp 287–293, Hitachi, 1988.

[5] S-H. Hsu and T. Chan, *A Study of Inspection Performance and the Personality of Quality Inspectors*, Int. J. of Quality & Reliability Management, Vol 11 No 5, 1994, pp 55–65.

[6] A. Martelli and U. Montanari, *Optimization decision trees through heuristically guided search*, Communications of the ACM 21(2), 1978.

[7] A.E. Maxwell, *Multivariate Analysis in Behavioural Research*, Chapman and Hall, London, 1977.

[8] R. Milne, *Real-Time Model Based Diagnosis of Gas Turbines*, in Research and Development in Expert Systems, M.A. Bramer and A.L. Macintosh (eds), pp 63–81, BHR Group Ltd, England, 1993.

[9] N.J. Nilsson, *Principles of Artificial Intelligence* Palo Alto, CA:Morgan Kaufmann, 1980.

[10] M.A. Rodrigues and Y.F. Li, *Dynamic Systems Modelling and Development of Robotic Applications in a Visual Programming Environment*, Robotics & Computer-Integrated Manufacturing, Vol. 12, No. 4, 1996, pp 311–320.

[11] M.A. Rodrigues, C. Loftus, M. Ratcliffe, and Y.F. Li, *Structure Notation of Dynamic Systems: a Pictorial Language Approach*, Proceedings of IEEE Computer Society International Conference on Computer Languages, Toulouse, France, May 16–19, 1994, pp 219 – 228.

[12] G. Taguchi, *Online Quality Control During Production*, Japanese Standards Association, Tokyo, 1981.

[13] J.D.T. Tannock, *Choice of Inspection Strategy Using Quality Simulation*, International Journal of Quality and Reliability Management, Vol. 12 No. 5, 1995, pp 75–84.

[14] J.S. Wiggins, *Cattell's System from the Perspective of Mainstream Personality Theory*, Multivariate Behavioural Research, Vol 19, 1994, pp 176–90.

[15] S.I. Yoo et al., *Fault diagnosis in Air Defense Artilleries Using Expert Systems*, Robotics and Computer Integrated Manufacturing, Vol 11, No 3, 1994, pp 177–186.

Identification and Defect Inspection with Ultrasonic Techniques in Foundry Pieces

A. Lázaro, I. Serrano, J.P. Oria*, C. de Miguel**

* Dept. Electronic Technology and Automatic Systems. University of Cantabria
E.T.S.I. Industriales y Telecomunicación.
Avda. Los Castros s/n 39005 Santander SPAIN
Tel: +34 42 201541 Fax + 34 42 201873.
e-mail: [lazaro, imelda, oria]@teisa.unican.es
** Environment and Workshop Management Group, FUNDIMOTOR S.A.
c/ José Mª Quijano s/n. Los Corrales de Buelna. Cantabria. SPAIN
Tel: +34 42 848900 Fax + 34 42 848914.

Abstract. This work describes an ultrasonic-based inspection and identification system for pieces. Given the well-known versatility and reliability of ultrasonic sensors for working in industrial environments, this system is developed for application to foundry pieces which will later be machined within the automobile industry. Starting from treatment of the ultrasonic signal reflected by an object and applying artificial intelligence techniques like neural networks, the identification of the different classes of pieces is carried out and the discrimination between machinable and non machinable pieces is made. In this field, ultrasonic tools appear to be a powerful technique for the quality control of processes because of their capacity to recognise and classify pieces. This work forms part of the investigation tasks of a collaboration project between the Engineering Control Group of the University of Cantabria and FUNIDIMOTOR S.A. (Nissan Group).

1 Introduction

In the manufacturing of foundry pieces, such as those used in the automobile industry, small defects can sometimes be found after casting like grains or small metal balls, and small notches. These defects, which are not critical for the final use of the piece, affect the manufacturing processes detrimentally, especially in machining tasks. This makes it necessary to inspect each piece at the end of the foundry stage with the aim of correcting the defects of the non-machinable pieces before going on to the machining stage. The progression of a non-machinable piece to this stage can cause damage to the machine tools that can complicate and increase costs in a relatively simple industrial process. The inspection system tries to solve this problem in an automated way applying ultrasonic identification techniques.

Ultrasonic sensors are attractive in industrial applications, because of their low cost and their versatility due to their reduced size and weight, and also their reliability and robustness in production environments, since they are able to work with darkness or variable illumination, dust, fumes and industrial noise [3]. Furthermore, ultrasonic techniques' are appropriate to be combined with other inspection techniques such as artificial vision, nowadays very widely used in the industrial world.

2 Identification Approach

The system will have to inspect any one of various foundry pieces, having to differentiate whether the piece under inspection is suitable or not suitable for the machining stage.

First of all, two types of mass-produced foundry pieces are going to be studied. These are foundry products for the manufacturing of brake drums and brake disks for automobiles.

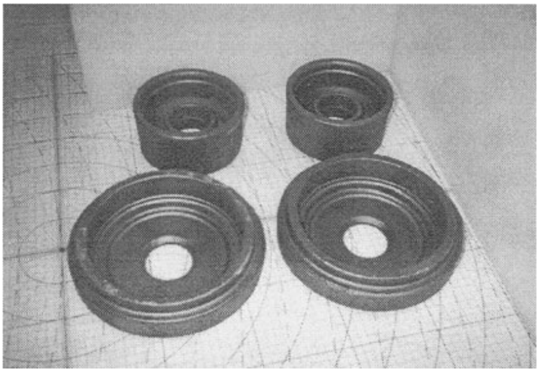


Fig. 1. Foundry pieces

Once the class of pieces is known, the selection of apt and inapt pieces must be done. The pieces which are considered apt for machining can have small differences with respect to a master. These differences mean that the ultrasonic echoes are not identical. The ambient conditions such as temperature and humidity also have an important role in this aspect. So, it is not easy to establish a threshold that allows us to distinguish between the echo of an apt piece and the echo of an inapt one, with respect to a master. It is necessary to reach a compromise in the estimation of the threshold between the machinable and the non machinable piece.

Figure 2 shows the different echoes of two foundry pieces of the same class, without significant defects and apt to be machined.

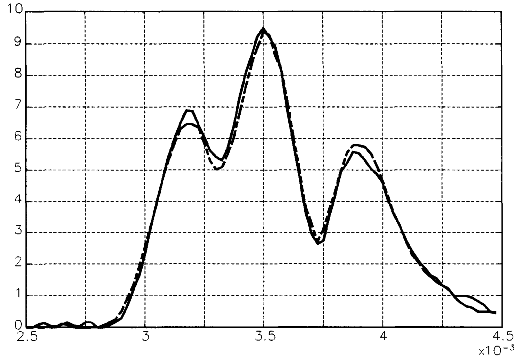


Fig. 2. Echoes of two foundry pieces of the same class

The system was trained with geometric shapes in order to develop and validate an identification technique that will later be applied to the foundry pieces for machining [5].

3 Ultrasonic Transucer

Among the extensive series of ultrasonic sensors which are available in the market, the sensor must be selected in order to fulfil the requirements of this specific application. Fundamentally, parameters such as the working frequency and the size of the sensor must be considered since the propagation cone of the sensor depends directly on these variables. This is important in order to achieve an optimal propagation cone and so to avoid dispersion problems [4]. For this application, a piezoelectric sensor MURATA MA4027, working at 40 khz has been chosen. The angle of dispersion of this sensor adapts well to the system and the tolerance of the sensor to industrial noise is also good. Furthermore, the transducer works in emisor-receptor mode avoiding problems of directionality and this does not affect the identification time negatively.

In this application a distance of 53 cm between the transducer and the piece has been used. The distance has been optimised to illuminate the piece and to obtain as much information as possible.

4 System

The transducer used is formed by an array with nine sensors distributed in the form of a 3*3 square matrix. The sensors are placed in a support whose dimensions are 75*75 mm. The sensors of the array work in emisor-receptor mode with the pulse-echo technique.

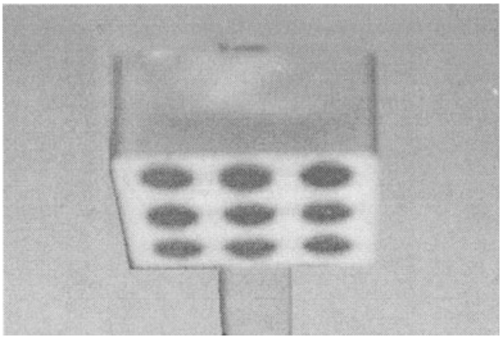
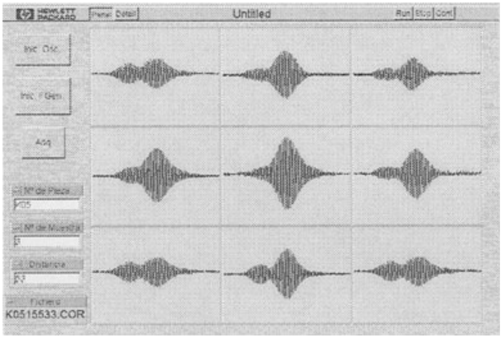


Fig. 3. Ultrasonic sensor array

This array is situated over the pieces to be identified which are illuminated vertically to collect the reflected echoes. The pieces are located in an ultrasonic anechoic surface, in order to avoid false echoes. For the generation and capture of the ultrasonic signal, HP-VEE visual programming has been used. This programming setting adapts to the applications of data acquisition and monitoring through the IEEE 488 standard and is oriented to production environments offering a good execution speed.

In the figure 4, the operation screen with captured echoes of each sensors of the array can be seen.



Nº de Pieza: 05

Nº de Muestra: 3

Distancia: 53

Fichero: K0515533.COR

Fig. 4. Operational screen

The sensors are excited with trains of pulses generated with a frequency of 40 khz. The ultrasonic signal generated is multiplexed in such a way that each sensor discharges and receives in an alternate way. It is possible to select independently the sensors to be used, since the use of all of them may not be necessary. The received echoes are demultiplexed and captured for their later mathematical treatment. After that, they are stored in libraries in the computer which directs and inspects all the acquisition operations, at the same time as performing the mathematical treatment.

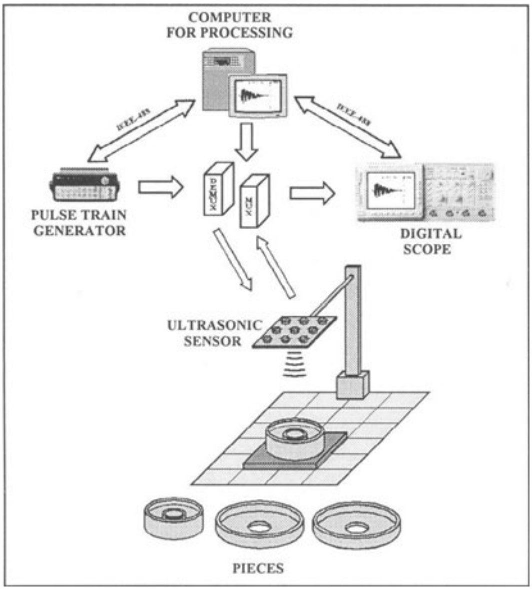


Fig. 5. Set-up

Although the system has been developed in an investigation laboratory, precautions with changes of temperature, humidity or air currents have not been taken in order to influence the robustness of a system which will have to work in an industrial world.

5 Mathematical Treatment

Once the echoes are obtained, the mathematical treatment stage, necessary for the identification, starts. The treatment begins with the detection of the envelopes of the echoes, which is carried out in digital mode, in contrast to other techniques which do this in a hardware way. This permits us to obtain greater filtering possibilities. Later, a decimation of this envelope is carried out and finally, a filtering which provokes the disappearance of the high frequency noise is carried out.

Once the initial treatment is concluded, distinct possibilities for the identification are considered. First, some echoes of machinable pieces are selected. These echoes will act as master patterns or references which will be compared with the rest of the echoes. The selection of echoes of pieces with typical heterogeneity has not been considered because the spectrum of possible defects is too extensive.

The correlation between the master echoes and the rest of the echoes is compared. A high correlation will imply that the echoes proceed from similar pieces, that is, the pieces are apt to be machined. If the correlation is low, therefore the pieces are different and have some defect and so they are non machinable [1]. The correlation formula used is

$$r^2 = \frac{\left(\sum xy\right)^2}{\sum x^2 \sum y^2} \tag{1}$$

and the elements of the signals compared are x and y.
After analysing the correlation index, the tolerance threshold for considering the piece apt or inapt for machining is established.

6 Identification through Neural Networks

Neural networks offer great guarantees and are very appropriate for identification tasks. There are a large numbers of neural net structures that can be applied to problems of artificial vision. Through a multilayer structure of perceptrons, with three layers of active neurons, any classification of patterns can be obtained [11]. For the ultrasonic recognition of objects, a completely connected three-layer feedforward neural net with back propagation learning has been used In this case, it has been necessary to take some patterns from echoes of non-machinable pieces as well as the echoes from machinable ones.

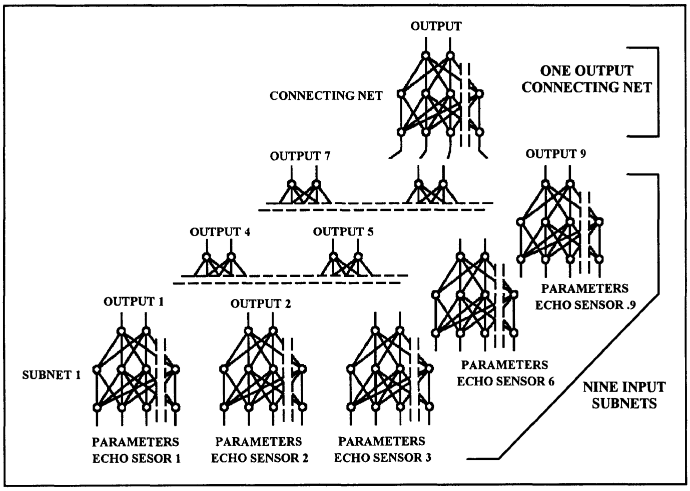
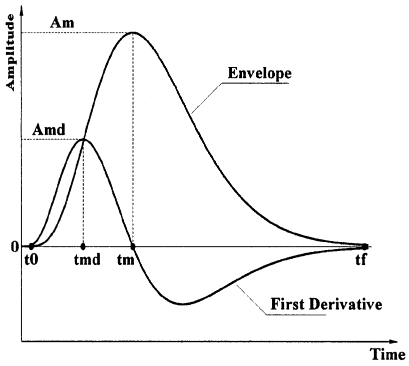


Fig. 6. Nerual Network

A set of nets with three layers in each has been used, one net for each sensor. The first layer of each subnet has a number of neurons equal at the number of parameters of each echo. The output layer has two neurons to indicate if the piece is apt or inapt for machining. The hidden layer can work with a variable number of neurons. All the output neurons are connected to a new connecting net, whose output layer provides the identification of the piece. The net has been trained by adding a quantity of 10% random variation to the input values of each learning cycle in order to improve the generalisation process [10]. To use the neural net, vectors of characteristic parameters of the envelope and the first derivative have been generated. To generate the characteristic vectors of the echoes, parameters of the envelopes and of the first derivative have been used. For each piece under test, five sets of echoes have been taken. The net was trained with echoes from known pieces and then checked with new sets of unknown echoes.



Am: Maximum amplitude of the envelope
tm: Time of the maximum Am
AE: Area under the envelope
AE1: Area under de envelope, between t0 and tm
AE2: Area under the envelope, between tm and tf
LE: Length of the envelope
ME: Ascending rate, defined as ME=Am/(tm-t0)
Amd: Maximum amplitude of the first derivated
tmd: Time of the maximum Amd
AD: Area under the first derivative
AD1: Area under the derivative, between t0 and tm
AD2: Area under the derivative, between tm and tf
LD: Length of the first derivative
MD: Ascending rate of the derivative, defined as MD=Amd/(tmd-t0)

Fig. 7. Characteristics of an echo

7 Results

* Identification results analysing the correlation index:

	Num. of patterns	Num. of Test Pieces		Threshold	Failures	**Percentage Success**
DRUMS	3	Mach.	3x5	0.97	0	100 %
		Non-mach.	3x15	0.97	0	
DISCS	3	Mach.	3x10	0.99	4	87 %
		Non-mach.	3x10	0.99	4	

Table 1. Correlation index information

* Identification results using neural nets:

Network Information	Network Parameters		Network Statistics	DRUMS	DISCS
# Layers	Learning rate	1	Number of Data Items	40	40
3	Momentum	0.9	Number Right	40	38
# Neurons per Layer	Input Noise	0.1	Number Wrong	0	2
9*(14/14/2)	Training Tolerance	0.05	Percent Right	100%	95%
1*(18/14/2)	Testing Tolerance	0.3	Percent Wrong	0%	5%
	Epochs per Update	1	Training Epochs (total)	2080	1277

Table 2. Neural Network information

Targets		
	N° Ident.	
	1	**2**
Drum M.	1	0
Drum U	0	1
Disc M.	1	0
Disc U.	0	1

Output		
	N° Ident.	
	1	**2**
Drum M. 1	0.9	0.0
Drum U. 1	0.1	1.0
Drum M. 2	0.9	0.1
Drum U. 2	0.1	0.9
Disc M. 1	0.9	0.1
Disc U. 1	0.1	0.8
Disc M. 2	0.6	0.3
Disc U. 2	0.2	0.8

Table 3. Output for four test sets of each type

With the correlation index technique, the identification can be performed, but the thresholds for distinguishing between machinable and non-machinable pieces are too high, and also they can only vary within a very narrow margin. Using the neural nets, the results are good. The failures in the identification of the discs could be solved but increasing the testing tolerance.

8 Conclusions

A promising technique has been developed for the identification and inspection of faults in foundry pieces. To completely validate the technique it is necessary to test it with a greater number of pieces of a wider range of classes. At the same time, it is also necessary to adapt the system to industrial environments in order to improve aspects like identification time, robustness, etc.

References

1. J.P. Oria, A. M. G. González, "Object Recognition Using Ultrasonic Sensors in Robotic Applications". Proceed. of the IECON'93 Hawai, USA
2. J.M. Martín,T. Freire, L. Calderón, "Ultrasonic Echoes from Complex Surfaces: an Application to Object Recognition", Sensors and Actuators, 1992
3. A. Lázaro, I. Serrano, J.P. Oria. "Sistema Ultrasónico de Detección y Reconocimiento de Piezas en Procesos Hostiles de Producción", XII Congreso Nacional de Ingeniería Mecánica. Bilabao, Spain, 1997
4. M.K. Brown, "Locating Object Surfaces with an Ultrasonic Range sensor", IEEE International Conference on Robotics, 1985
5. I. Serrano, A. Lázaro, J.P. Oria. "A New Method for Object Identification in Ultrasonic Systems Using Neural Nets", CIRA'97. Monterey, California, USA, 1997
6. M.Wybow, C. Wykes, "An Ultrasonic System for Object Recognition in a Manufacturing Environment", Journal of Intelligent Manufacturing, 1992
7. Lazaro, I.Serrano, J.P. Oria, "Robot-Sensor System Integration for 3D Object Recognition in Ultrasonic Images", ACRA`97. Tokyo, Japan., 1997
8. F. Ramírez, et al., "Introducción a los Métodos de Ensayos no Destructivos de Control de Calidad de los Materiales", INTA.
9. L Fausett, "Fundamentals of Neural Networks", Prentice Hall Int. De., 1994
10. H. Demuth, M. Beale, "Neural Network Toolbox for use with Matlab",The MathWorks, Inc. 1992-1996
11. Beale, T. Jackson, "Neural Computing", Adam. Hilger, 1990.
12. J Krautkramer, H. Krautkramer, "Ultrasonic Testing of Materials", Springer-Verlag, 1990.

An OOM-KBES Approach for Fault Detection and Diagnosis

Yunfeng Xiao and Chia Y. Han

Department of Electrical & Computer Engineering and Computer Science
University of Cincinnati, Cincinnati, Ohio, USA 45221-0030
chia.han@uc.edu

Abstract. This paper presents an integrated approach to the intelligent building research: using both the object-oriented modeling (OOM) and knowledge-based expert-system (KBES) methodologies and technologies for fault detection and diagnosis (FDD) of building HVAC systems. The approach consists of five basic steps: 1) establish a model simulating the behavior of the target system using object-oriented design methodologies; 2) identify all types of faults in the target system, extract rules for each process to build the knowledge bases; 3) integrate the knowledge bases into system model to allow the system perform FDD task on itself; 4) build an on-line monitoring system to collect all real-time setpoint data; and 5) make inference against the knowledge bases based on real time data and generate reports.

1 Introduction

Given that failures in physical components are unavoidable, prompt fault detection and diagnosis (FDD) is a must in many engineering systems. FDD of large electro-mechanical systems, such as the HVAC (heating, ventilating, and air-conditioning) systems in buildings, is an important part of maintaining proper performance, reducing energy consumption, and increasing the system reliability and availability [1], [2]. A single fault in a large control system may result in a large number of nested symptoms, and the situation will be even worse when there are multiple concurrent faults, so the major difficulties come from isolating the primary source of failure. With the growing sophistication and complexity of electro-mechanical systems, it becomes increasingly difficult to detect and isolate system faults.

How to find faulty condition and identify the fault is the major issue of the FDD research in many industries. In the building industry, FDD in HVAC system has been explored as part of an International Energy Agency collaborative research project [2]. Recently, many artificial intelligence (AI) methodologies and technology have been applied to perform fault detection and diagnosis on HVAC system and applications have shown that they have an impact within the HVAC industry [3, 4, 5, 6, 7]. However, they have been applied mostly on single unit of the HVAC system or a component, like the air handling unit (AHU). Our objective is to build a FDD system on the entire HVAC system.

A typical process flow of a large VAV (variable air volume)-based HVAC system consists of seven major processes, which include, as represented in Figure 1, the following: (a) Mixed Air, (b) Preheating, (c) Cooling, (d) Humidifying, (e) Supply Air Flow Control, (f) VAV box, and (g) Return Air Flow Control.

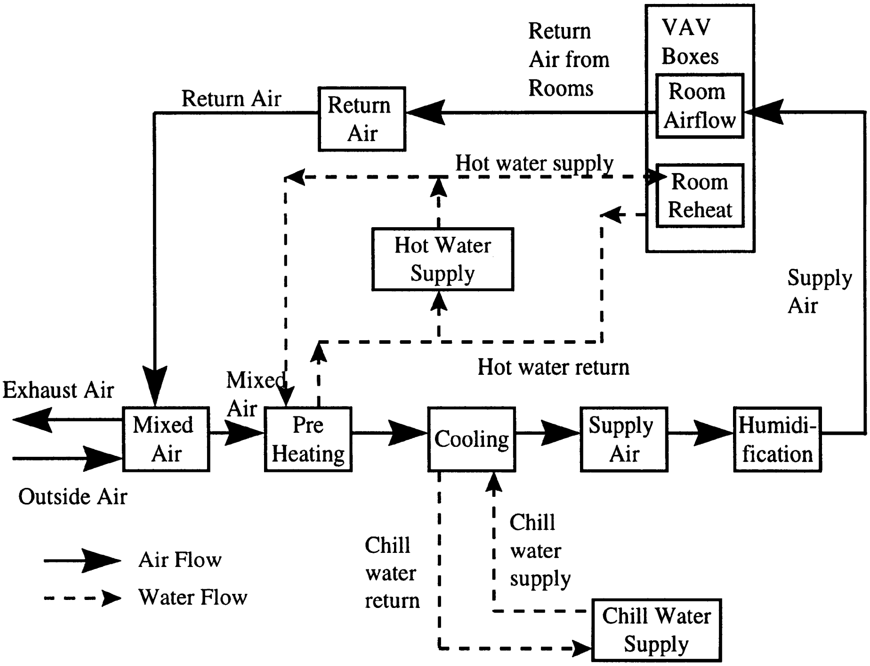


Fig 1. HVAC System Flow Chart

Currently, most of the modern and large commercial building's HVAC systems use direct digital control (DDC), i.e., the use of microprocessor-based controller, along with the electric sensors and actuators, to implement sophisticated control equations and algorithms to provide the best combination of comfort and efficiency. The best FDD approach is to build a FDD module on top of the existing programmable control based on both the real-time (low-level) data collected by the DDC system and the system control logic (high-level knowledge) embedded in the control program. We shall present an approach that uses both object-oriented modeling (OOM) and knowledge-based expert system (KBES) methods for implementing the HVAC's FDD solution. In Section 2, a model of the system components, processes, and classes definition are given. Section 3 will present how the knowledge for FDD is extracted, represented, and used in fault detection and diagnosis. A conclusion is given in Section 4.

2 System Model - An Object-Oriented Modeling Approach

By using the object-oriented modeling techniques, we model building HVAC systems. The OOM system model is then used as the framework in which a knowledge based expert-system is embedded to perform the FDD tasks.

As shown in Figure 2, the system consists of four modules: System Monitor, System Model, Fault Identifier, and user interface/reporting facility. Firstly, System Monitor makes the HVAC operational data available to the system model. It collects real time data from a collection of setpoints, filters out the noise and stores them in database. Secondly, System Model modularizes the HVAC system. It abstracts the HVAC system into a hierarchical architecture so that the FDD operations can be performed over those modules at each layer and then integrate the results to get the system-wide FDD result. Such a layered structure breaks down the complexity of HVAC and makes it easier to handle. Thirdly, GUI/FDD Report facility coordinates the interactions between the FDD system and users. It takes user's instructions and returns back the FDD result both in graphical and textual report forms.

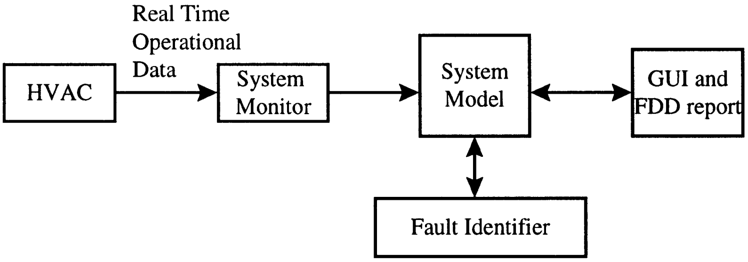


Fig. 2. The Procedures of Fault Detection and Diagnosis

Lastly, Fault Identifier uses knowledge based expert system to identify faults in every HVAC process. This module actually consists of eight submodules, seven process sub-modules for all the seven HVAC processes and a system level sub-module for the entire HVAC system. All these sub-modules are embedded in our system model. In this paper, our main focal points are the System Model and Fault Identifier modules.

The HVAC electro-mechanical system model is divided into three layers: the system layer, the process layer, and the component layer. Each layer is composed of different classes, which encompass various objects in lower layer. Figure 3 shows the structure. The system layer is an overall envelope that encapsulates the entire HVAC system. There will be only one class, the system class in this layer.

The system class provides the basic functionality and system specification of a HVAC system and acts as an interface between lower layers and those applications that need to access them. Also included in this class are those process objects from the second layer.

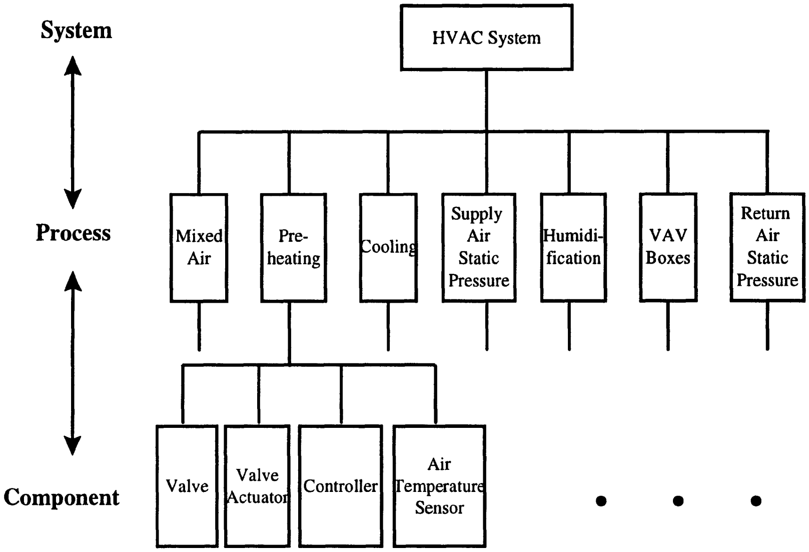


Fig. 3. 3-Tier System Model: System-Processes-Components

Figure 4 shows the class definitions and the relationships. Based on different design requirements and application purposes, the system layer class will decide which processes are included, and all together decide the behavior of the HVAC system.

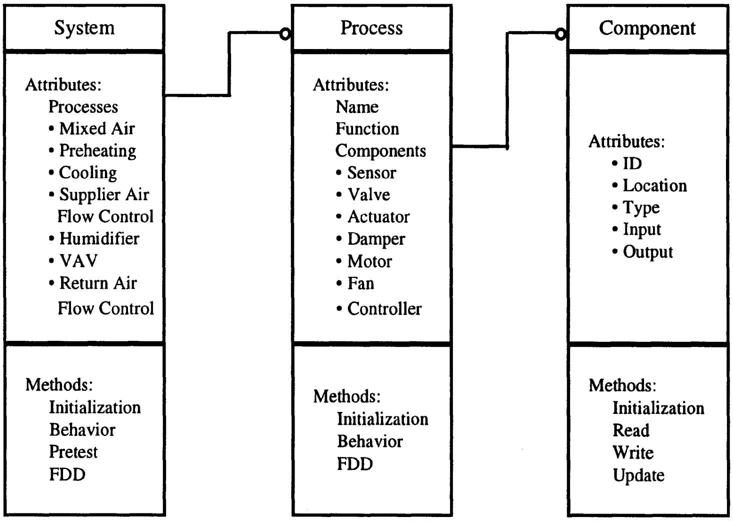


Fig. 4. Classes Definition and Relationship

A Pretest method is included to set up the basic valid testing conditions by verifying that all the sensors as well as the external systems such as pumps and both hot and chilled water supply sources are working fine before invoking the FDD methods in lower level to test all the processes and integrate the results to generate the system FDD reports.

The process layer includes all the processes that were mentioned in the last section. Each process represents a function of the HVAC system. The output of one process could be the input of another. Linking all the processes, as shown in Figure 5, will form the entire HVAC work flow. This layer is the interface between the system layer and component layer. Similarly, the objects from the component layer constitute classes in this layer. Different processes decide which components are included, and all components will decide the functionality and behavior of the process.

The last layer, the component layer, consists of the most fundamental elements, the component classes. All the basic physical components of the HVAC will be defined in this layer. Sensors, actuators, valves, dampers, and controllers are all included in this layer.

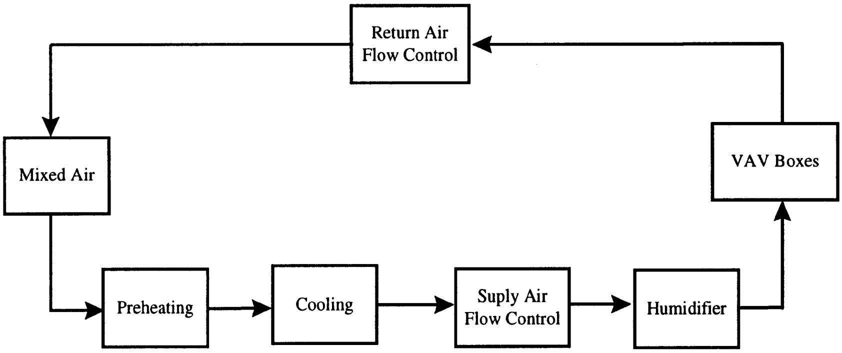
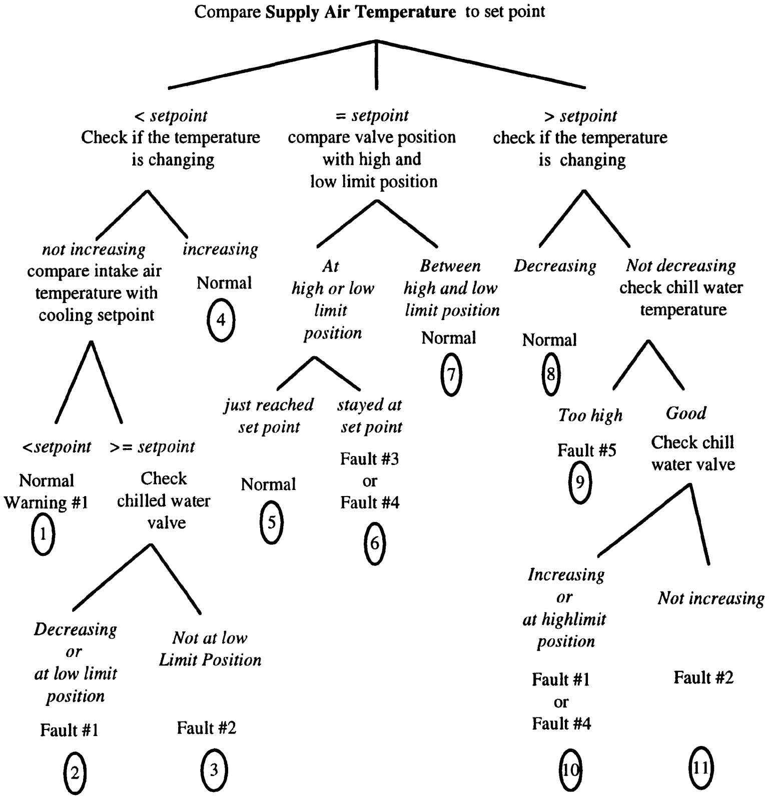


Fig. 5. HVAC Process Work Flow: MA-PH-CL-SF-HM-VAV-RF-MA

Different component combinations constitute different physical process compositions. Almost all faults will eventually be at one or several of these components. The health of a process will depend mostly on the health of its components, and likewise, the health of the entire system will depend on the health of each process.

3 Fault Identification - A KBES-based Approach

We used a knowledge-based, expert system (KBES) methodology and technology to identify the faults. Since our target systems are direct-digital controlled (DDC), and the logic of the programs of the controller, normally represented in a process control language (PCL), contains all the operating knowledge of the system, any violation of the DDC program is considered a fault. Our knowledge base is built based on this principle. The logic of the DDC program can be represented as a decision tree which could be readily converted into If-Then rules in a rule-based expert system.


Compare **Supply Air Temperature** to set point
< setpoint
Check if the temperature is changing
= setpoint
compare valve position with high and low limit position
> setpoint
check if the temperature is changing
not increasing
compare intake air temperature with cooling setpoint
increasing
Normal
4
At high or low limit position
Between high and low limit position
Normal
7
Decreasing
Normal
8
Not decreasing
check chill water temperature
<setpoint
Normal
Warning #1
1
>= setpoint
Check chilled water valve
just reached set point
Normal
5
stayed at set point
Fault #3
or
Fault #4
6
Too high
Fault #5
9
Good
Check chill water valve
Decreasing or at low limit position
Fault #1
2
Not at low Limit Position
Fault #2
3
Increasing or at highlimit position
Fault #1
or
Fault #4
10
Not increasing
Fault #2
11

Fault #1: Chilled water valve, actuator or positioner failure.
Fault #2: Controller - Program or output-transducer failure.
Fault #3: Controller bad tuning.
Fault #4: Chilled water delivery problem: clogged, strainer, air in coil, or closed valve.
Fault #5: Building lever chilled water problem: chiller, chilled water pump problem.

Warning #1: Check air temperature set point.

Fig. 6. FDD Decision Tree for the Cooling Process

The general guideline for extracting rules from DDC program consists of the following steps: 1) Analyze the DDC program; 2) draw decision tree that covers all cases that could possible happen; 3) extract general rules from decision tree; 4) convert general rules to its formal form and put the rules into knowledge base(s). The cooling process is used as an example to illustrate the rule extracting procedure.

By analyzing the cooling control program, eleven different cases, which are shown as terminal nodes in the decision tree in Figure 6, are identified. In Node 2, for instance, the control goal is not achieved because the discharge air temperature is lower than its set point and is not increasing. The input of this process, the mixed air temperature, is higher than the cooling process set point. Under this circumstance, the cooling process should lower the air flow temperature to its set point. But at this node, the process over lowered the air temperature. The controller should now close the chilled water valve to bring the temperature back to its set point. After checking the valve control signal, we know that the controller is trying to close the valve. It's clear that the valve or its actuator didn't follow the control signal, indicating a valve or actuator failure. Therefore, a fault is identified and the FDD result in this case is "Fault#1". The rule and its formal form corresponding to this case are illustrated in Figure 7 as follows.

IF	*the discharge air temperature is lower than set point*	IF	`D04DT < D04DSP`
	and		`and`
	the discharge air temperature is NOT increasing		`NOT IsIncreasing(D04DT)`
	and		`and`
	Mixed air temperature is higher than set point		`D04MT > D04DSP`
	and		`and`
	The controller is trying to close the valve		`(IsDecreasing(D04DLO) or IsAtLowLimit(D04DLO))`
THEN	FDD result is (Fault #1)	THEN	`CoolingResult=(Fault#1)`

Fig. 7. The mapping of a branch of the decision tree (left column) to an IF-THEN rule (right column), where D04DT, D04DSP,... are the real point names in our target HVAC system.

The similar procedure can be repeated for all the other 10 nodes of the cooling process, and likewise for all other processes. Thus, the knowledge base can be built in an incremental manner. We may add more knowledge to both refine and enhance the decision tree. However, if we develop the decision tree one level deeper, we will generate more rules, this may trade off the accuracy of FDD with the processing time.

In our three-layer system model, there is a FDD method in each class in the upper two layers. These FDD methods can perform fault detection and diagnosis on each object of the classes. By implementing our fault identifier as these FDD methods, we integrated the System Model and Fault Identifier together. The system layer FDD function makes FDD request to each of the lower layer processes, which invokes its own FDD function and passes the FDD result back to system layer FDD method. After getting all the FDD results from the lower layer processes, the system FDD method generates the system FDD result and feeds it back to the user.

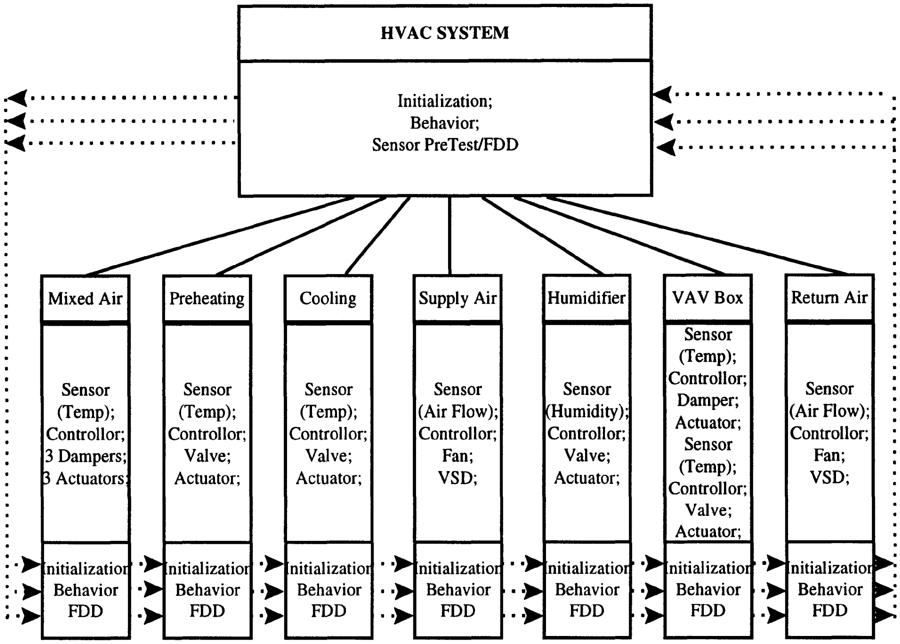

HVAC SYSTEM
Initialization;
Behavior;
Sensor PreTest/FDD
Mixed Air
Sensor (Temp); Controllor; 3 Dampers; 3 Actuators;
Preheating
Sensor (Temp); Controllor; Valve; Actuator;
Cooling
Sensor (Temp); Controllor; Valve; Actuator;
Supply Air
Sensor (Air Flow); Controllor; Fan; VSD;
Humidifier
Sensor (Humidity); Controllor; Valve; Actuator;
VAV Box
Sensor (Temp); Controllor; Damper; Actuator; Sensor (Temp); Controllor; Valve; Actuator;
Return Air
Sensor (Air Flow); Controllor; Fan; VSD;
Initialization
Behavior
FDD

Fig. 8. FDD System Model and Overall Work Flow

Figure 8 shows the system integration and FDD work flow. The Initialization function initializes all component objects by invoking the object's own initialization function. This function decides the configuration and the functionality of the HVAC system being modeled. The Behavior function decides the system-wide behavior by invoking the process object's behavior functions and linking them together in the order of process cycle shown in Figure 8. This linking is done by feeding the output of one process into the next process downstream. The FDD function performs the PreTest function and all FDD functions of those process objects. PreTest functions can be thought of as the preparation of FDD and are invoked inside FDD function. Their main purpose is to check the status, as well as the accuracy, of the sensors as the data collected from the sensors will drive the FDD system. Any invalid sensor will implicate the entire reasoning and decision making process.

4 Conclusions

This paper presented a HVAC's FDD system based on a new approach, integrating the object-oriented modeling and the knowledge-based expert system methodologies, as well as the various off-the-shelf client/server technologies. The System

Monitoring module that interfaces with the HVAC control systems was done in C++ and the collected real-time set point data were stored in a MS-Access data base. The FDD Module was implemented using a commercial expert system shell, M4. The graphical user interface and report module was implemented with Visual Basic.

Unlike other works reported in the literature, our major contribution is in building an on-line FDD system targeting the whole real HVAC system rather than a partial system or a component in a laboratory setting. The system has been implemented on real buildings on our campus, a relatively large size university campus. The expert system approach allows an incremental refinement of system operating knowledge. It also allows us to customize the various configuration easily. The design is based on the open-system architecture. Future technologies can be easily integrated for better performance. As the system model is a scaleable object-oriented framework, it is generic and very suitable for adding any new functions and other physical building components.

References

1. W. Y. Lee, J. M. House, C. Park, G. E. Kelly, "Fault Diagnosis of an Air-Handling Unit Using Artificial Neural Networks," ASHRAE Transactions, Vol.102 (1996)
2. http://www.ecbcs.org/annex25.html, "Real Time Simulation of HEVC Systems for Building Optimization, Fault Detection and Diagnosis," IEA, (1997)
3. M. Stylianou and D. Nikanpour, "Performance Monitoring, Fault Detection, and Diagnosis of Reciprocating Chillers," ASHRAE Transactions, Vol.102 (1996)
4. H. Yoshida, et. al., "Typical Fault of Air-Conditioning Systems and Fault Detection by ARX Model and Extended Kalman Filter," ASHRAE Transactions, Vol.102 (1996)
5. H. Peitsman and V. E. Bakker, "Application of Black-box Models to HVAC Systems for Fault Detection," ," ASHRAE Transactions, Vol.102 (1996)
6. A. L .Dexter and M. Benouarets, "A Generic Approach to Identifying Faults in HVAC Plants," ASHRAE Transactions, Vol.102 (1996)
7. Charles C. Culp, *The Impact of AI Technology within the HVAC Industry,* ASHRAE Journal, Volume 32 Number 12 (1990)

Model-Based Fault Simulation: A Reduction Method for the Diagnosis of Electrical Components

Andreas Heinzelmann[1] and Paula Chammas[1]

[1] Daimler Benz AG, Research FT2/EK T721
D-70546 Stuttgart, Germany
{heinzelm, chammas}@dbag.stg.daimlerbenz.com

Abstract. Today's systems, for example motor vehicles, can be characterized by a variety of subsystems and components. For the automatic investigation of all possible breakdowns and the consequences thereof, an entire system simulation is inevitable. Without taking particular measures into account a variety of individual simulations are necessary in order to copy all the electrical error conditions of the subsystems or components. The necessary computing times for this are astronomical. In the following article a method is described which makes it possible to restrict the number of times the simulation is carried out to a practical number prior to the simulation.

1 Introduction

Certain features have to be taken into consideration in order to simulate the faulty behavior of an entire system. These include the control of the individual electronic components, the component conditions describing faulty and normal behavior, the operating conditions and the observable or measurable system variables. A system condition is characterized by a particular combination of features. If one takes all combinations into consideration it becomes easy to recognize the complexity of this problem. It comprises of an exponential growth in the number of the components and the control conditions. In the relevant available literature, for example in [2], this problem is generally known. The goal is to restrict the number of conditions, on whose base the entire simulation is to be conducted, to the relevant simulation cases. With this a shorter simulation time is achieved and the resources for the diagnosis knowledge is limited to the necessary quantity.

In the following description of the *Reduction Method for the Diagnosis of Electrical Components* (REMDEC) electrical harnessing data is used as input information and allow the necessary simulation cases to be ascertained prior to the whole system simulation. The model described is unique in that an optimal system reduction occurs in the electrical periphery of the control units. This is a result of the reduction method that is based on the regularity of the electrical network theory. The process described is applicable to different simulation tools. The methodology represents a further development of the research carried out by *Mauss* [4].

2 Description of the Electronic Components

Resistors, connectors, wires, plugs, lamps, wire bundles, sensors, relays, fuses etc. are the fundamental construction elements that appear in the periphery of the control units. This information is generally available along with the structural relationships in a technical data processing form, for example in CAD systems. As a result of the practical relevance, the following considerations concentrate mainly on the passive periphery. Active components can be analyzed in the same fashion but they play a subordinate role in the area described above. A truly passive periphery can be represented by a set of electrical resistance with different values. Current and voltage sources as well as resistances are apparent in active circuits. These primary components will be referred to as *basic components.*

Behavior of the components is characterized by discreet conditions, for example disconnection and short circuit. Discreet values are present in the basic components that represent the different electrical conditions of the components. Discreet conditions of the basic components are in a particular relationship to the component conditions that comprises of the normal behavior and breakdown behavior of a component. It is possible to model the periphery of the control units (CU) as a graph without direction in which the basic components with the discreet values are represented by the edges *GE*. In order to connect the basic components, connection points are necessary that appear in the graph as nodes. In general, the behavior of the electronic periphery characterizes the basic periphery graph G_{ep}.

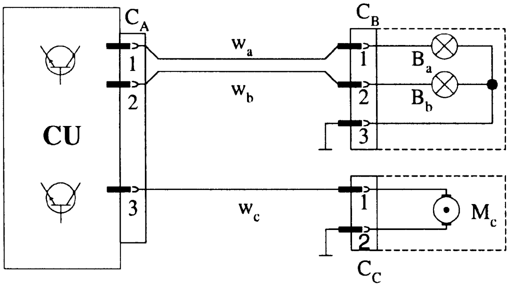


Fig. 1. Control Unit with Periphery

In a basic peripheral graph G_{ep} the node potential P_i or the currents I_i can be measured. The measurability of the node potential P_i or the branch current I_i are dependent on the local situation and the structure of the whole system. Every node potential P_i and every branch current I_i can take on the value of *measurable* or *not measurable.* Furthermore the potential P_i on the node N_i or a branch current I_i contains a known or unknown value. The periphery of the control units can be represented as a graph without direction. To illustrate this point figure 1 shows a small periphery of a control unit. Figure 2 portrays the basic periphery graphs G_{ep} that belong to the principle circuits represented in figure 1. The controlling electronics of the CU are modeled in the basic peripheral graph G_{ep}.

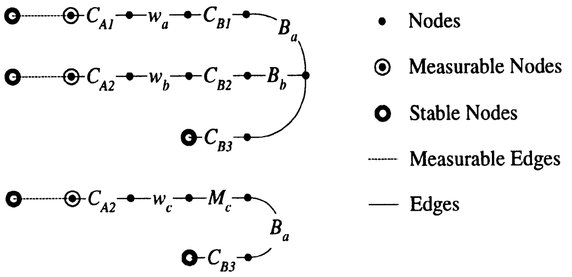


Fig. 2. Basic Graph of the Periphery of the Control Units

3 Reduction of the Basic Periphery Graph

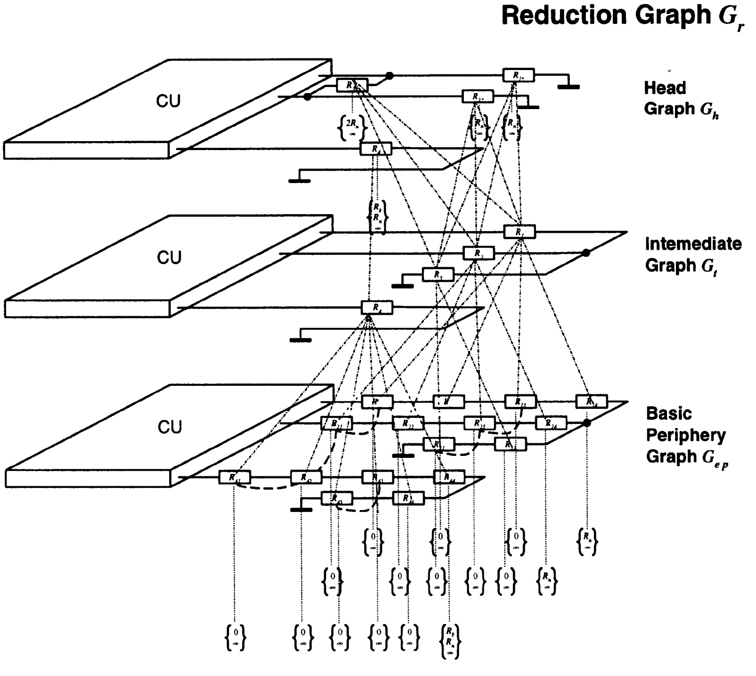
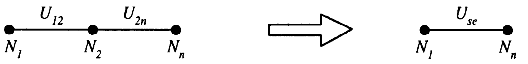


Fig. 3. Reduction Graph of a Control Unit Periphery

On the basis of the periphery graph introduced above, G_{ep}, a graph reduction can be generated in which the fundamental characteristics of behavior with respect to the observable physical variables are contained. Figure 3 shows the reduction graph G_r of the explained example. It consists of a basic periphery graph G_{ep}, one or more intermediate graph G_i and one head graph G_h.

In order to build the reduction graph, well known base operations from the network analysis [3,6] are used. The basic consideration that stands behind the process is that under certain conditions branches in an electrical network can be represented by an equivalent network with a simpler structure. For the efficient construction of a reduction graph equivalent operations, as well as individual red operations are necessary. The network structure is transformed by equivalent operations into a representational form which portrays the output point for a further reduction step. Below are three reduction operations for passive elements as examples. One must take into consideration that behind the basic components there is not only one individual value but a set of possible values.

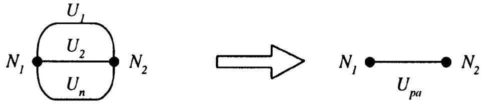
Series Reduction



$$V_{se} = \sum_{i=1}^{n-1} V_{i(i+1)} \text{ for } \textit{Resistance } V = R \tag{1}$$

Series reduction is applicable for the basic components of the type with the resistance R. For these basic components, U_i, the values V_k are possible.

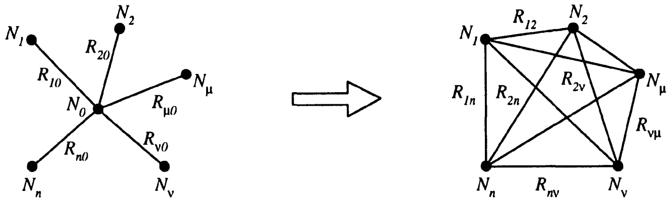
Parallel Reduction



$$V_{pa} = \sum_{i=1}^{n} V_i \textit{ for Resistance } R = 1/V \tag{2}$$

The above reduction process is performed for the type of basic component with the resistance R.

Star-Reduction



Calculation of the resistance values is carried out using the following equation [3]:

$$R_{\nu\mu} = R_{\nu 0}\, R_{\mu 0} \sum_{m=1}^{n} \frac{1}{R_{m0}} \tag{3}$$

Ascertaining the condition of the entire system via simulation using the control of the components and the component conditions is the primary aim. The quantity of all system states is proportional to the permutation of the component conditions. Discreet values represent components states. Permutations of the discreet values V_{ij} from the basic components describe the whole or part of the system condition. These discreet values are represented by the values of the graph elements *GE*. In order to copy the entire behavior of a sub-graph G_{sub} into a upper graph G_{upp} it is necessary to generate the permutation between the discreet values V_{ij}.

A so-called connecting-operator is introduced in order to restrict the permutations from non realistic values of combinations. For example, the Star-Reduction, by connecting the discreet values from different components creates a permutation block from the connecting-operator. The reduction step calculates the values by considering all connections in the permutation block. As many desired combinations of the discreet resistance values are allowed to be taken into consideration during this calculation process of the reduction operation. Moreover, other non desired combinations are excluded in advance. Therefore, for example, only single faults in the electrical periphery are specified and reduced by this process. Figure 4 demonstrates this calculation process.

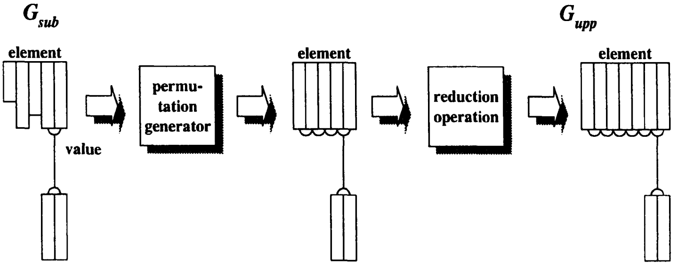


Fig. 4. Calculation process with a permutation block

3.1 Algorithms for the Creation of a Reduction Graph

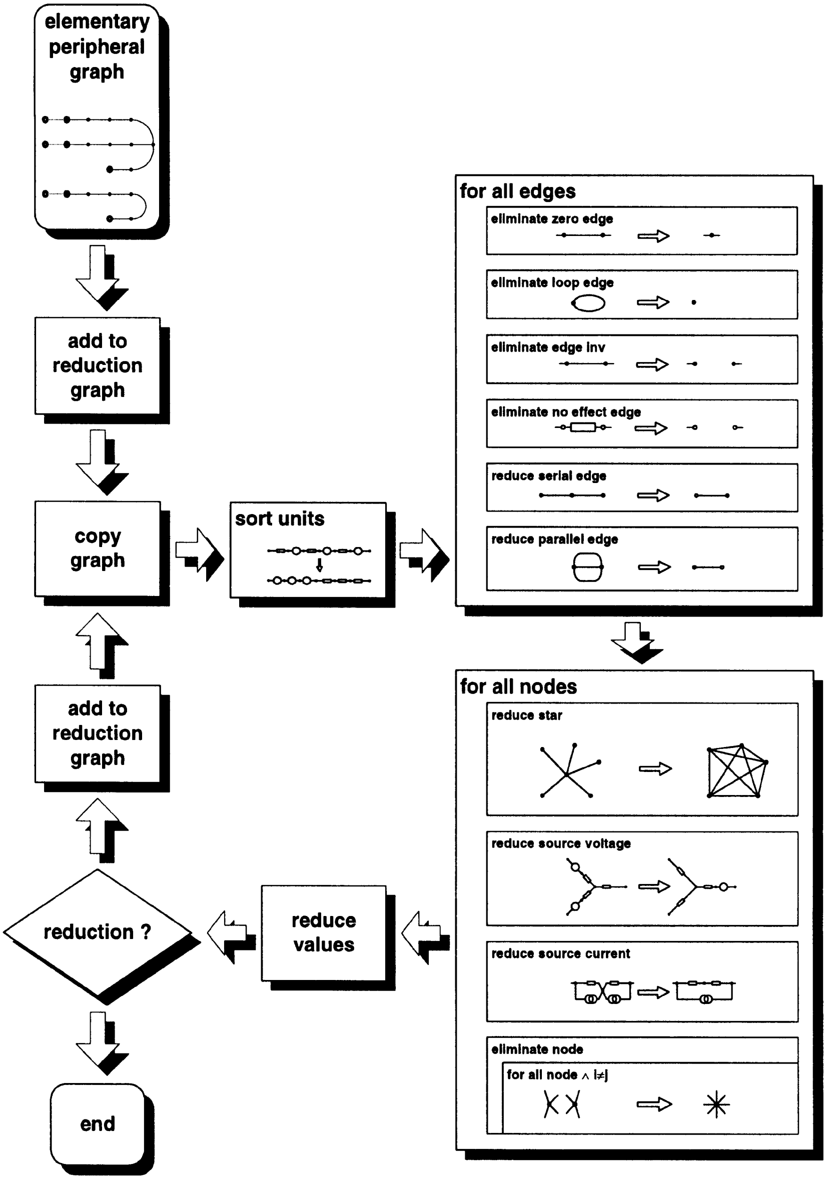


Fig. 5. Reduction Process for Electrical Circuits

Figure 5 shows how the reduction process functions. A basic graph without direction $G_{ep} = (U, N)$ with nodes N_i and the basic components U_i as edges is a starting point. As a result of the reduction process the specific electrical characteristics contained in the base graph are simplified with regard to the measurable variables. According to *Mozetic* [5] the reduction occurs on the following three levels:

- Summary of similar discreet values
- Distancing from the graph elements that are not relevant for the diagnosis
- Simplification of the original structure via replacement structures

3.2 Creation of Regions

The basic idea behind the creation of regions is that not all operating and breakdown conditions of components have an effect on all system variables. There are large independent areas affected by the particular conditions of components and areas that are not affected. In order to minimize the resources used by the simulation it is advisable to identify such areas with their elements. Creation of regions makes it possible to reduce a system wide problem to the affected elements in the system and therefore limit the entire diagnosis problem to a confined local area.

Using a graph traversing process the related areas are recognizable. In differing literature sources, for example in [8], one can find algorithms used for traversing graphs. The borders between the regions are nodes, N_i, with fixed potential. Creation of regions generates a list illustrating the region to which the basic components belong. As a result of the creation of regions, a complex basic periphery graph is split into confined areas. The areas that are present in one graph element show no relationship to graph elements of other areas. Every region makes up a independent part-problem and can therefore also be characterized as a focus area.

4 Representation of the discreet Conditions on the Component

After a simulation is completed the results thereof must be again assigned to the component conditions. The necessary information can be made available in table format. For every region a relationship-table of all possible component conditions and the corresponding discreet values of the basic components is created. The table can also be restricted to the selected combination of component conditions. During the entire reduction process the table is continually updated. At the end of the reduction process the component conditions that were specified at the beginning are to be found in the table with the discreet values of the head graph.

With help of the relationship table one can ascertain the associated components from the head graph for a known discreet value with relative ease. If one adds the simulation results next to the last column of the relationship-table it results in the classic fault-table with the component conditions and the corresponding system variables.

5 Conclusion

Currently, the method described above is being implemented as a prototype with CAD data serving as input data. On the output side of the reduction process there is the simulation and diagnostic tool roδon [7] that ascertains the values of the process variables during a simulation.

The method described here demonstrates a possible process for carrying out complete system simulation of electrical components in the future. Current simulation and diagnostic tools hardly allow the portrayal of individual part systems and often fail when the scope is extended to the entire system. *REMDEC* allows a complex system to be reduced to a substitute system without losing information that is relevant to the diagnosis in the process of the model. Following a preliminary investigation the reduction factor moves in an exponential dimension. The process can be implemented independent of a tool and can be used for the creation of diagnostic systems, for example, decision trees or declarative lists as well for the simulation of entire systems.

References

1. Akl, Selim G.: The Design and Analysis of Parallel Algorithms. Prentice Hall, New Jersey (1989)
2. de Kleer, J.; Williams, B. C.: Diagnosing Multiple Faults. In: Hamscher, W., Console, L., de Kleer, J. (eds) Readings in Model-Based Diagnosis. Morgan Kaufmann Publishers, San Mateo CA (1992) 100-117
3. Küpfmüller, K.; Kohn, G.: Theoretische Elektrotechnik und Elektronik, Springer-Verlag, Berlin (1993)
4. Mauss, J.; Neumann, B.: Qualitative Reasoning about Electrical Circuits using Series-Parallel-Star Trees. In Tenth International Workshop on Qualitative Reasoning, AAAI Press (1996) 147-153
5. Mozetic, I.: Hierarchical Model-based Diagnosis. In: Int. J. of Man-Machine Studies, 35(3), 329-362
6. Philippow, E.: Taschenbuch Elektrotechnik, Band 1: Allgemeine Grundlagen. VEB Verlag Technik, Berlin (1986)
7. roδon 1.6a, Reasoning Over Systems in their Entirety, Das modellbasierte Diagnose- und Simulationssystem. Handbuch Band I und II. ROSE Informatik, Heidenheim.
8. Sedgewick, R.: Algorithms in C; Addison-Wesley, Bonn (1992)

A Model-Based Automated Diagnosis Algorithm

Along Lin

Extended Enterprise Laboratory
Internet Business Management Department
Hewlett-Packard Laboratories
Filton Road, Stoke Gifford
Bristol BS12 6QZ, U.K.
Tel: +44 0117 9229413 Fax: +44 0117 9229250
Email: alin@hplb.hpl.hp.com

Abstract. The explosive growth of the Internet and the World Wide Web has made management tasks such as configuration, monitoring, diagnosis, and fault correction much more complicated. An automated management system is highly needed to alleviate those tasks of system managers. Some tools use Case-Based Reasoning or Rule-Based Reasoning to pinpoint faults. However, they have difficulties in solving problems requiring deep level knowledge. Although some Model-Based diagnostic systems have been developed, they are mainly used to tackle problems in other domains. In this paper, a goal-driven model-based automated diagnosis algorithm is proposed.

Keywords: Model-Based Diagnosis, Management, Automation.

1. Introduction

The explosive growth of the Internet and the widespread acceptance of the World Wide Web have made the management of networks and systems much more complicated. Several intelligent tools are highly required to fundamentally alleviate management tasks such as configuration, monitoring, diagnosis and fault correction, which are knowledge intensive and require a great amount of expertise. An automated management tool will play a very important role in providing customers with high quality services over networks because of its lower maintenance cost and less disruptions.

Model-Based Reasoning (MBR) has been widely researched and accepted as the principal diagnostic technique in electronic circuit analysis, power station maintenance and medical diagnosis domains [3-10]. However, little emphasis has been put on its application to network and system management domain. In MBR, it is assumed that the structure of a diagnosed system and its behaviors have been well understood and the basic principles about how its components behave and their causalities have been described in its models. We can use that knowledge to diagnose the faults in a component by comparing its observations with the predictions about its intended behaviors based on the models. Because it can diagnose the faults which have not been pre-determined, it has been considered as a more promising technique than other knowledge-based methods such as Rule-Based Reasoning (RBR), Bayesian Networks (BNs), Neural Networks (NNs), and Case-Based Reasoning

(CBR). A Model-Based diagnostic system consists of a fault detection mechanism and a diagnostic engine. Some autonomous systems can also include a recovery mechanism. In MBR, all what we need to do is to write the models of all domain components. Each component is described in terms of its behaviour and relationship to other components.

FLIPPER is a prototype automated management system which manages devices like workstations, Unix, Netware servers, PC's, printers, routers. It is based on Logic Programming technology and like a former prototype Dolphin, also developed by Hewlett-Packard Laboratories, Bristol. It consists of a model compiler, an inference engine. It has a typed declarative modeling language which supports a simple Object-Oriented inheritance. It works based on imported models for managed components in a system. Some DLLs are also provided to access and check the information about those managed objects. To manage systems in a distributed environment, FLIPPER can also act as an intelligent agent, residing on a managed device. FLIPPER agents can interact and collaborate with each other based on Inter-Agent Protocol.

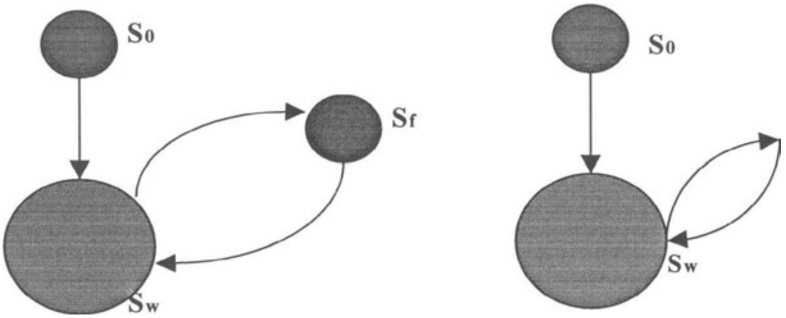
In FLIPPER, a component is modeled by an object whose functionality is determined by methods. FLIPPER models define object types in a managed domain, express the relationships among objects, and describe external functions linking objects to the real world to access and check information about managed components. All the domain knowledge is captured in the models. We usually think of the models in MBR as causal links or deep level knowledge necessary to provide explanations of faults. In FLIPPER, while an inference engine is used to prove goals, faults are identified by a diagnostic engine.

In the following, we present a managed system's life cycle first. Then, a Goal-Driven Model Based Diagnostic Algorithm (GDMBDA) is presented in detail.

2. The Life Cycle of a Managed System

Systems must be configured or installed correctly by using some tool before it can work. Then, the system will be in a consistent state. Unfortunately, during its run-time, there are some events occurred, causing the system to enter some incorrect states. An automated management system is needed to alleviate system administrator's task of maintaining the system in a correct state more efficiently. Reactive and proactive management are two major types of system management. The reactive one manages a system by fixing faults after diagnosing a problem, whereas the proactive approach tries to prevent a system from being in a faulty state by monitoring and reconfiguring a system. Both approaches need a mechanism to detect a fault or some tendency to reach the thresholds set for some sensitive parameters in advance. In Fig. 1, **S0, Sw, Sf** represents the initial state, working state and faulty state of the managed system, respectively. The state transitions from **S0** to **Sw** and from **Sf** to **Sw** are accomplished by configuring the managed system initially and by repairing the diagnosed faults or reconfiguring the system. Therefore, automated fault diagnosis is crucial to reactive system management. Obviously, it is the repairing that closes the loop and makes it possible to automatically maintain the managed system in a consistent and desired state.

In FLIPPER, goals are used to describe the relationships among several managed components. Administrators use them to express their management intentions. There are two special types of goals. Monitored goals are always watched and used to trigger diagnostic engine or generate an event to signal some state change in the system to a system administrator. Therefore, the fault-detection mechanism can be accomplished based on monitored goals. Resident goals need to be maintained true, which involves monitoring and fixing. As a resident goal becomes false, a sequence of executable primitive actions will be executed automatically. After executing them, the system will be in a consistent state in which the failed resident goal becomes true. In FLIPPER, those primitive actions or scripts have been pre-compiled in advance. However, we believe a better solution is that they should be generated by a knowledge-based system or an intelligent planner based on models of the managed system [1].



(a) Reactively managed system which has fault states in its state space.
(b) Proactively managed system which has no fault states.

Fig. 1. The two main types of life cycles of a managed system

3. A Goal-Driven Model Based Diagnostic Algorithm

The rules in the models could be expressed by Guarded Horn Clauses. For an accurate diagnosis, some semantic information is provided in the domain object models. A rule ***R*** is expressed as: ***H*** IF ***G*** | ***B***, where ***H*** , ***G*** , and ***B*** are the head, guard and body of ***R***, respectively. Facts are the special rules whose bodies are empty. The sub-goals in ***B*** can be divided into two categories: diagnosable goals and non-diagnosable goals. Model writers are responsible for attaching the semantic tag "diagnosable" to the rules which they think are meaningful to the real diagnoses. A goal is diagnosable if one of the rules defining it is diagnosable or the body of one of those rules contains a diagnosable goal. A guarding goal ***G*** must not contain any diagnosable goals. For a fault diagnosis, ***G*** does not contribute to any faults in a system, it is just used to commit to some rule ***R***. Although backtracking is allowed within ***G***, ***B*** is deterministic. GDMBDA consists of algorithms **3.1~3.5** detailed below.

In order to describe GDMBDA more concisely, we introduce following notations:
NT(N) returns the type of a node *N* in a search tree, its domain is {***and***, ***or***, ***leaf***};
T(G) returns the attribute of a goal *G*, its domain is {***true***, ***false***, ***unknown***, ***ignored***}. If a goal ***G*** has been proven true/false, ***T(G)*** becomes ***true/false***; ***T(G)*** has ***unknown*** as default. If ***G*** is not interested in any more, ***T(G)* = *ignored***,.
CP(N) is the choice point of the goal represented by node ***N***, pointing to *N*'s candidate rules. ***CP(N)*** becomes ***NULL*** if *N* has been fully expanded.
PN(N) denotes the parent of ***N***. For a root node ***N***, ***PN(N)*** is ***NULL***.

```
Algorithm 3.1 Expand(N, DE)
1 Let next = NULL; If (NT(N) == or) {R = CP(N); go to 7;}
2 CP(N) = NULL;
3 If N is an access method, call its corresponding procedure to get or check the
  attributes of a managed objects in the real world.
  If it fails, return(Propagate(N, false, DE)); else return(Propagate(N, true, DE));
4 If there are no rules defined for N, return(Propagate(N, false, DE));
5 Let R1, ... , Rn be the n rules defined for the goal corresponding to N, they are
  stored in a linked list of which R is the head; Let expand = false; go to 7;
6 R = R→ NEXT;   // backtracking point
  If (R == NULL)
      {If (expand ≠ true) return(Propagate(N, false, DE)); return(next);}
7 Let H, G, B are the head, guard and body of R, respectively.
  // $\theta = \theta 1 \bullet \theta 2$, where $\theta 1$ is inherited from N whereas $\theta 2$ is the current unifier;
  If there is no Most General Unifier (mgu) $\theta$ such that $N \bullet \theta == H \bullet \theta$, go to 6;
  If G is empty, go to 9;
8 If Prove($G \bullet \theta$) returns false, go to 6;
  // test whether N should be committed to R
9 If (expand == false) expand = true;  // N can be expanded
  If (NT(N) == leaf) NT(N) = or;
10 Construct a node C, PN(C) = N, $\theta$ is attached to C;
11 If (B is empty) {if (next == NULL) next = Propagate(C, true, DE); }
     else  {Let b1, ... , bm be the sub-goals in B;
              construct nodes t1, ... , tm, consisting of $b1 \bullet \theta$, ... , $bm \bullet \theta$, respectively;
              For j = 1 ,..., m   // RB links the sub goals in a rule body
                 {T(tj) = unknown; NT(tj) = leaf; PN(tj) = C; If (j < m) RB(tj) = tj+1;}
              NT(C) = and; T(C) = unknown;
              If (next == NULL) next = t1;
            }
12 go to 6.
```

In algorithm **3.1**, **Prove** is the inference engine, which has the backtracking mechanism. For the sake of efficiency, we can expand ***N*** by one rule at a time, expand others later. This can be achieved by changing step 12 to 12':

12' ***CP(N) = R→ NEXT***. Return(*next*);

Actually, for any fully bound goal, only one of the rules defining it can be committed to. In this case, after committing to a rule, the choice point of a goal is ***NULL***. Thus, step 12 can be changed to:

12'' Return(*next*);

Furthermore, the search tree of ***(G•θ)*** has been dropped from that of ***C***. This is because for a real fault diagnosis ***G*** does not contribute to any faults in a system, it is just used to commit to some rule ***R***.

```
// G is a leaf node which has been proven to be true or false;
Algorithm 3.2 Propagate(G, value, DE)
1  NT(G) = leaf; T(G) = value;
2  Let N = PN(G) and assume N has children N1, N2, ... , Nn;
3  Let E= T(G); F = NT(N);
4  If (E == ignored || F == or && E ==false) || F == and && E== true))
   go to 7;
5  call IgnoreChildren(N);
   // N's children can be ignored if we just want to find single fault
6  T(N) =E;
   If DE is diagnose and G is non-diagnosable, let T(N) = T(G) = ignored;
   go to 9;
7  If (F == or && CP(N) ≠ NULL) return N; // N has not been fully expanded;
8  If (T(N1) == T(N2) == ... ==T(Nn)) // N has been fully expanded;
      {T(N) = T(G); go to 9; }
   T(N) = ignored; // Test whether there are unknown nodes in N's children
   For j =1 ,..., n,
      if (T(Nj) == unknown) T(N) = unknown;
9  If (PN(N) == NULL) return N;
   If (T(N) == unknown)
      {If (F == or) return N; else return(RB(G));}
10 Let G = N, go to 2.
```

```
Algorithm 3.3 IgnoreChildren(N)
1  If (T(N) ≠ unknown) return;
2  If (NT(N) == leaf)
      { if (T(N) == unknown) T(N) = ignored; return;}
3  Let N1, N2, ... , Nn be N's children;
   For j =1 ,..., n, IgnoreChildren(Nj);
   return;
```

Both the diagnostic tree and the explanation tree of a goal ***G*** can be constructed from its search tree. The diagnostic tree of a failed goal is denoted by ***DT(G)***.

```
Algorithm 3.4 Construct(G)
1 If (T(G) == ignored) return NULL;
2 G is in DT(G);
  If (NT(G) == leaf) return G;
3 Let Gi (i =1, ... , n) be n children of G and E = T(G). If (NT(G) == or) E = ~E;
4 If E is false
     { there must be some Gj  ($1 \le j \le n$)  such that T(Gj) = T(G),
       DT(Gj ) belongs to DT(G);
     } else { for i =1, ... , n, if (Construct(Gi) $\neq$ NULL) DT(Gi ) is in DT(G);}
5 Return DT(G).
```

The diagnostic tree of a failed goal *G* is returned by Diagnose(*G*):

```
Algorithm 3.5 Diagnose(G)
1 Construct a leaf node N, corresponding to the failed goal G;
  Let T(N) = unknown; PN(N) = NULL; NT(N) = leaf; N' = N;
2 While (T(N) == unknown) N = Expand(N, diagnose);
3 Return Construct(N').
```

Thus, the diagnosis of the failed goal *G* consists of all the leaf nodes in Diagnose(*G*). If there are multiple faults in a system, after fixing the diagnosed faults by executing the sequence of primitive actions, generated by a planner, we can diagnose others similarly. Alternatively, we can extend GDMBDA to find all faults.

4. An Example of Model-Based Diagnosis

In some companies, information retrieval is strictly controlled by company's policies. Assume some company adopts Role-Based Access Control (RBAC) to protect its confidential information from any unauthorized accesses. With model based diagnosis, we first must describe the models of RBAC, which means the role of a user determines what permissions s/he can have. Usually, a user is assigned to some particular role which is associated with some permissions or privileges. Each privilege specifies an operation on some classes of data. In the following example, User, Operation, Info Role, Key, Privilege, Opcration, Machine, Level are user defined object types. List is a built-in primitive type. Info has sub-types such as File, Directory, URL, Data-Base, and Network-Device, which has Printer, Machine as its sub-types.

```
[User u] authorized [Operation op] on [Info i] (+ +) DIAGNOSABLE IF
    [i] classified [Level l] & // URL is a sub-type of Info
    // each piece of information is classified as one level and has a place to store it
    [u] assigned [Role r] &    // the user r has been assigned to some role r
    [r] permitted [List privileges] & // the role r has been given a list of privileges
    [privileges] member [Privilege p] & // get an authorized privilege for role r
    [p] operates [op] on [List levels] &
    // obtain the operation and information levels a privilege p can perform
    [l] within [levels]  // test current information sensitivity is allowed user u
    // none of above relations or access methods is a DIAGNOSABLE goal
    |
    [u] canAccess [i].  // user u must be allowed to access the information i
[User  u] access [URL url] (+ +) DIAGNOSABLE IF
    [url] service ["http"] site [String site] port [Integer p] page [String name] &
    [webService w] machine [site] port [p] & // get the web server
    [w] maps [name] to [NTFile file] & //  mapping  web page name to a file name
    [w] running & // check whether web server Netscape is running
    [u] machineName [String name] &
    [Machine m] has [name]  // get the user's machine name
    |
    [w] allows [m] access [file].
[webService w] allows [Machine host] access [NTFile file]
    (+ + ?) (+ + +) DIAGNOSABLE  IF
    [w] sub-Keys [Key k] var ["name"] value ["default"] &
    [w] defaultAllows [host] access [file] with [k] &
    [file] pathName [String path] &
    [w] sub-Keys [Key _] var ["ppath"] value [path] & // get registry information
    [w] sub-Keys [Key s] var ["ppath"] value [path]
    |
    [w] allows [m] access [path] with [s].
[webService _] defaultAllows [Machine _] access [NTFile _] with [Key k]
    (+ ? + +) (+ + ? +) (+ + + +)  IF
    ~[k] contains ["DirectiveName"] equals ["PathCheck"] at [Key _].
[webService w] defaultAllows [Machine host] access [NTFile file] with [Key k]
    (+ ? + +) (+ + ? +) (+ + + +)  IF
    [w] allows [m] access [file] with [k].
[Role r] permitted [List privileges] (+ ?) IF
    [r] has [privileges]. // empty body
[Role r] permitted [List privileges] (+ ?) IF
    [r] isa [Role r1] & // The privileges of roles can be inherited
    [r1] permitted [privileges]. // empty body
```

When a user failed to access a particular web page, he can use the diagnostic engine using the GDMBDA to identify the underlying causes. Based on the models for a web page access, we could find out the user is unauthorized to access because of the web page control or the IP address or DNS of user machine is disallowed by the web server. Because guarded Horn Clauses are used and only diagnosable relations are present in the search tree, a lot of irrelevant information will be eliminated in the final diagnosis.

5. Conclusions

Fault diagnosis is a very important issue in an automated management system. MBR appears to be a more promising technique than other knowledge-based methods such as Rule-Based Reasoning, Case-Based Reasoning, Bayesian Networks, and Neural Networks because it can diagnose the faults which have not been predetermined. Faults in a system can be diagnosed automatically based on the models, which describe the normal behavior of the system. However, because Model-Based approach reasons from the actual structure and function of a system, it is inefficient for some problems. Furthermore, obtaining domain models is sometimes either difficult or too complicated, whereas most of the faults can be diagnosed based on past experience, which is very effective if the rule base or the case library is either comparatively small or well-indexed. A better solution is a hybrid approach integrating some of the above mentioned approaches. A rule base can be used to control the choice of the fault candidates in MBR, and to decide when to use different methods for a specific fault diagnosis. A case library will be provided to access the solutions to some fault diagnoses occurred previously, of which the domain models are unavailable. For some diagnoses, their solutions and contexts can also be stored in the case library for reuse later. Frequently occurring faults in a system can be diagnosed efficiently even by a few of heuristic diagnostic rules. We believe that such a hybrid diagnostic approach will perform better than any of them does.

Further researcher include:

- Implement the proposed algorithm in a Logic Programming language supporting Guard Horn Clauses or an extended FLIPPER modeling language;
- propose a hybrid approach to the fault diagnosis in an automated system management;
- develop an efficient planner which can generate a sequence of primitive actions automatically to rectify diagnosed faults, configure/re-configure a system [1];
- integrate fault detection, monitoring, diagnosis and fault correction into an automated management system;
- add security mechanisms to FLIPPER so that it can manage systems remotely;
- generate component models in a system automatically from its descriptions used to its design and development by using CAD/CAM technologies.

In the future, we will apply the approach described in this paper to manage some systems and practical applications such as services on Internet, security of information retrieval, business processes, and so on.

Acknowledgements

I would like to thank following people for their help: Alan Hydes, Claudio Bartolini, Adrian Baldwin, Masoud Mansouri-Samani, Giuliano di Vitantonio at Hewlett-Packard Laboratories, Bristol, United Kingdom. I also wish to acknowledge the anonymous reviewers of this paper for their helpful comments.

References

1. Lin, A.: A Logic-Based Approach to Automated System Management, In *Proc. of the 6th Int. Conf. on Practical Applications of PROLOG and the 4th Int. Conf. On Practical Applications of Constraint Technology (PAP/PACT-98)*, March 1998.

2. Baldwin, A., Bartolini, C., Vitantonio, G.D., Eshghi, K.: A Novel Algorithm for Fault Diagnosis in Internet-based Services, *Workshop on OVUA*, 1997.

3. Chen, J.S. and Srihari, S.N.: Candidate Ordering and Elimination in Model-Based Fault Diagnosis, In *Proc. Of the 11th IJCAI*, 1363-1368, 1989.

4. Davis, R.: Diagnostic reasoning based on structure and behaviour, *Artificial Intelligence,* 24(1), 347-410, 1984.

5. Hofmann, M.O.: Model-Based Diagnosis Directed by Heuristic Search, *The Ninth Conference on Artificial Intelligence for Applications*, 197-203, March,1993.

6. de Kleer, J.: Focusing on Probable Diagnoses, In *Proc. 9th National Conf. On Artificial Intelligence,* 842-848, July 1991.

7. de Kleer, J., Mackworth, A.K., Reiter, R.: Characterizing Diagnosis and Systems, *Artificial Intelligence,* Vol. 56, 197-222, 1992.

8. de Kleer, J.and Williams, B.C.: Diagnosing multiple faults, *Artificial Intelligence,* 32(1), 97-130, April 1987.

9. R. Reiter, A Theory of Diagnosis From First Principles, *Artificial Intelligence*, 32(1), 57-96, 1987.

10. Sakao, T., Umeda, Y., Tomiyama, T.and Shimomura, Y.: Model-Based Automatic Generation of Sequence-Control Programs from Design Information, *IEEE Expert*, 54-61, 1997.

Cyclic Forecasting with Recurrent Neural Network

Shaun-inn Wu

Department of Computer Science
California State University
San Marcos, CA 92069
USA
shauninn@csusm.edu

Abstract. General statistical method such as the Box-Jenkins ARIMA(p,d,q) model have long been applied in forecasting. Statistical methods such as auto-regression has been used as an efficient and accurate way for forecasting in certain applications such as stock-market forecasting. However, one still has to monitor the forecasting system and determine whether to adjust the parameters to reduce forecasting errors when applying auto-regressive method. A recurrent neural network has been designed to make the forecasts of auto-regression. Then the weight adjusting strategies of the recurrent neural network can be used to continuously adjust the parameters based on the forecasting errors. Therefore, we obtain the forecasts efficiently based on auto-regression without having to monitor the forecasting system constantly and adjust the parameters manually. This provides a very effective tool in forecasting monthly cyclic trends in importing and exporting in a harbor.

1. Introduction

Most of forecasting systems are based on statistical methods such as Box-Jenkins methodologies of time series [1]. Recently, artificial neural networks have been applied to forecasting and have achieved certain success [2], [3].

When the general Box-Jenkins methodologies of time series are applied, the tasks of tentative model identification, parameter estimation and diagnostic will have to be performed before the model can be used to forecast. It is computationally expensive to handle large volume data, especially in stock-market forecasting [4]. However, in order to have a more accurate prediction, these systems sometimes are required to forecast based on the data of the last ten or twenty years. Therefore, most of such systems can only forecast based on monthly data of past years to be computationally feasible. Moreover, whenever a new time series of historical data is presented, Box-Jenkins forecasting methodology has to be applied to identify the model and estimate the parameters to produce future forecasts. Because the model identification can only be judged by the researchers from the available Auto-Regressive Integrated Moving-Average or ARIMA(p,d,q) candidate models, this is a labor-intensive task.

We developed several stock-market forecasting systems based on competitive backpropagation neural networks [5]. The systems were trained with the S&P 500 composite indexes of past twenty years, produced the forecasts and adjusted themselves by comparing their forecasts with the actual indexes. The systems were able to forecast based on much more fine-grained daily data without much computational cost. We also implemented some statistical systems based on Box-Jenkins ARIMA(p,d,q) model of time series and the simple exponential smoothing. Their system parameters were previously reported [6]. In general, neural networks outperform pure statistical systems. A recurrent neural network was designed to emulate simple exponential smoothing without having to manually adjusting its parameter [7]. The system has been enhanced to emulate auto-regression with the advantage of automatically adjusting all the system parameters.

We then applied our recurrent neural network on forecasting the cyclic trends in the commodity volumes in a harbor [8]. The system has been trained with twenty years of commodity volumes from 1971 to 1991 in the port of Kaohsiung, the biggest harbor in Taiwan. It was shown that the recurrent neural networks successfully identify the cycles in the past data to assist in the future forecasts.

2. Auto-Regression

Not all of traditional statistical forecasting methods require as much work as Box-Jenkins methodology of ARIMA(p,d,q) time series model. An interesting statistical technique called *Simple Exponential Smoothing* has been an effective forecasting tool in financial forecasting [9]. It can iteratively and efficiently produce forecasts. It is actually equivalent to *Exponentially Weighted Moving Average* model or ARIMA(0,1,1) model. If we can assume that the time series, Y_t, fluctuates about a gradually changing mean level, then a reasonable model for Y_t is $Y_t=ß_t+ \varepsilon_t$ where $ß_t$ is the mean of Y_t which is time dependent and ε_t is white noise at time t.

Let Y_1, Y_2, ..., Y_n denote observed values of the above time series. An estimate for $ß_t$ is simply the sample mean, $\hat{ß}_t = \bar{Y}$ which implies that each value of the series is equally important to the estimation of $ß_t$. However, since it is assumed that $ß_t$ is gradually changing, applying equal importance to past observations may not be reasonable. It may be more advantageous to apply heavier weights to current observations and declining weights to observations far into the past to reflect the change in $ß_t$ [10]. Simple exponential smoothing (SES) is such a forecasting technique in which unequal weights are assigned to the observed data to provide updated estimates of the model parameter $ß_t$. Let $\hat{ß}_{t-1}$ be the estimate of $ß_t$ at time t-1, t=1, 2, ... where $\hat{ß}_0=Y_1$. At time t, $\hat{ß}_{t-1}$ is updated to $\hat{ß}_t$ by using the smoothing equation $\hat{ß}_t = \alpha Y_t + (1-\alpha)\hat{ß}_{t-1}$. The value of the smoothing parameter α is in the interval (0,1) and Y_t is the observed value of the time series at time t. If the estimate for $ß_t$ at time t is $\hat{ß}_t$, then the forecast at the point Y_{t+1} is simply $\hat{Y}_{t+1} = \hat{ß}_t$ and one can rewrite the smoothing equation as $\hat{Y}_{t+1} = \alpha Y_t + (1-\alpha)\hat{Y}_t$. The advantage of this method is that it is computationally simple and yet produces satisfiable forecasting results [11]. A more general extension of simple exponential smoothing is the Auto-Regressive model.

Assume a time series data $Y_1, Y_2, \ldots, Y_N$ has no trend, no seasonal variations, and that there is a stationary process which generated the data with the mean μ_t as a constant and the auto-covariance $r(s,t) = \mathrm{Cov}(Y_s, Y_t) = \mathrm{Cov}(\varepsilon_s, \varepsilon_t) = \sigma^2$ if $t = s$ or 0 if $t \neq s$. The Auto-regressive model of order p, AR(p), can be defined as follows: ε_t, $t=0, \pm 1, \pm 2, \ldots$ is a sequence of independent identically distributed random variable with mean 0 and variance σ^2. $Y_t = \varepsilon_t + \phi_1 Y_{t-1} + \ldots + \phi_p Y_{t-p}$, $t=0, \pm 1, \ldots$.

However, one still has to monitor this forecasting system to determine whether the weights need to be adjusted to reduce forecasting errors. Since artificial neural networks can provide weight adjusting algorithms, it can be used to provide a auto-regression technique with an adaptive weighting scheme.

3. A Recurrent Neural Network

In auto-regression, the values of ϕ_i's are chosen in such a way as to minimize mean square errors of forecasts. A usual method of finding appropriate values of ϕ_i's is to approximate the partial autocorrelations. Once a set of values for ϕ_i's is determined, it is incorporated into the function to generate future forecasts. Since no forecasting system will provide perfect forecasts, it is essential to monitor the system in some fashion to determine whether an adjustment of ϕ_i's is necessary. Hence, the method of auto-regression still runs into the problem of manually monitoring and adjusting the system.

The auto-regression can be emulated by an auto-regression recurrent neural network (ARRNN) as in the following structure:

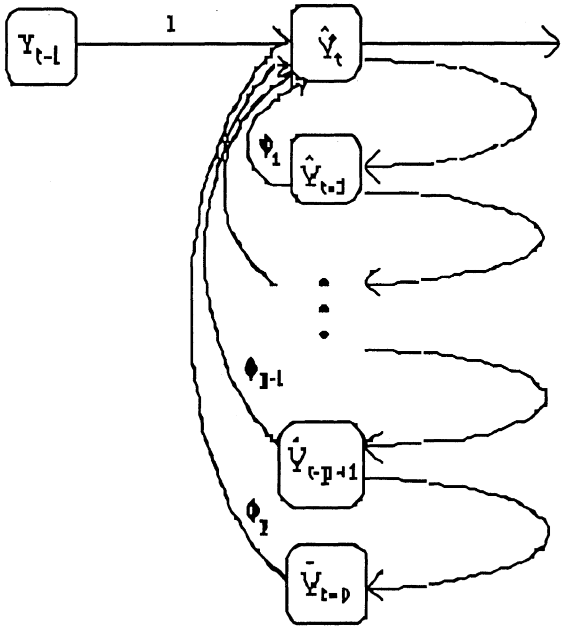


Fig. 1. The recurrent neural network produces the forecasts of auto-regression.

It is easy to understand that the above recurrent neural network will produce almost the same forecasts as the technique of auto-regressive method as long as the value of ϕ_i's are known.

In our neural network as shown in **Fig. 1**, the parameters become the weights of the connections between the input node Y_{t-1} and the past p outputs of output node $\hat{Y}_t$. The weight adjustment formula of the neural network is used to monitor the parameters of auto-regression. Since the neural network is self adjusting its weight connections, it does not need *manual monitoring and adjusting*. Although these two systems are similar to each other, the neural network has the advantage of *automatically* adjusting itself to the new data and simultaneously producing forecasts.

4. Mathematical Modeling of ARRNN

Given the index for the day t-1, Y_{t-1}, the estimate for the index at the day t is

$\hat{Y}_t = Y_{t-1} + \phi_1 \hat{Y}_{t-1} + \ldots + \phi_{p-1} \hat{Y}_{t-p+1} + \phi_p \hat{Y}_{t-p}$.

Let $w_0 = 1$, $w_i = \phi_i$ for i = 1, 2, . . ., p, we have $\hat{Y}_t = w_0 Y_{t-1} + w_1 \hat{Y}_{t-1} + \ldots + w_{p-1} \hat{Y}_{t-p+1} + w_p \hat{Y}_{t-p}$.

We will minimize the mean square errors of the forecasts as follows:

$$F(w_0,w_1,...,w_{p-1},w_p) = \frac{1}{N}\sum_{t=1}^{N} (\hat{Y}_t - Y_t)^2 .$$

For weight adjustment, we calculate the gradient of $F(w_0,w_1, . . .,w_{p-1},w_p)$, $\nabla F(w_0,w_1, . . .,w_{p-1},w_p)$. We can derive $\frac{\partial F(w_0,w_1,...,w_{p-1},w_p)}{\partial w_i}$ = $\frac{2}{N}\sum_{t=1}^{N} (\hat{Y}_t - Y_t) R_i(t)$ where $R_i(t) = \frac{\partial \hat{Y}_t}{\partial w_i}$ for i = 0, 1, . . ., p-1, p.

Since $\frac{\partial \hat{Y}_t}{\partial w_i} = Z_i + w_1 \frac{\partial \hat{Y}_{t-1}}{\partial w_i} + . . . + w_{p-1} \frac{\partial \hat{Y}_{t-p+1}}{\partial w_i} + w_p \frac{\partial \hat{Y}_{t-p}}{\partial w_i}$

for i = 0, 1, . . ., p-1, p where $Z_0 = Y_{t-1}$ and $Z_i = \hat{Y}_{t-i}$ for i = 1, . . ., p-1, p, we have $R_i(t) = Z_i + w_1 R_i(t-1) + . . . + w_{p-1} R_i(t-p+1) + w_p R_i(t-p)$ for i = 0, 1, . . ., p-1, p.

Therefore, $R_i(t)$ can be computed based on the values of Y_{t-1}, $R_i(t-s)$, $\hat{Y}_{t-s}$ and w_s at day t-s for s = 1, 2, . . ., p.

Let $E_i(t) = \frac{\partial F(w_0,w_1,...,w_{p-1},w_p)}{\partial w_i} = \frac{2}{N} \sum_{t=1}^{N} (\hat{Y}_t - Y_t) R_i(t)$

for i = 0, 1, . . ., p.

Then $\nabla F(w_0,w_1,...,w_{p-1},w_p) = (E_0(t), E_1(t), ..., E_{p-1}(t), E_p(t))$ and $E_i(N)$ can be computed based on $R_i(t)$, Y_t, $\hat{Y}_t$ at day t and hence Y_{t-1}, $R_i(t-s)$, $\hat{Y}_{t-s}$ and w_s at day t-s for s = 1, 2, . . ., p.

Since $E_i(N) = \frac{2}{N}\sum_{t=1}^{N}(Y_t - \hat{Y}_t) R_i(t)$, we have

$$E_i(N+1) = \frac{2}{N+1}(Y_{N+1} - \hat{Y}_{N+1}) R_i(N+1) + \frac{N}{N+1} E_i(N).$$

Hence, the value of $E_i(N+1)$ can be calculated based on the values of $R_i(N+1)$, Y_{N+1}, $\hat{Y}_{N+1}$ and $E_i(N)$ and $\nabla F(w_0,w_1, . . .,w_{p-1},w_p)$ at day t can be calculated based on $\nabla F(w_0,w_1,...,w_{p-1},w_p)$ at day t-1.

The weight vector $W(t) = (w_{0(t)}, w_{1(t)}, \ldots, w_{p-1(t)}, w_{p(t)})$ can be computed by $W(t) = W(t-1) - \beta \nabla F(w_0, w_1, \ldots, w_{p-1}, w_p)$ where β is a learning factor.

5. Conclusion

We have shown that auto-regression can be emulated by a recurrent neural network. We have also shown that the weights of such a neural network can be easily adjusted. Hence, the recurrent neural network can produce the forecasts of auto-regression without having to manually monitoring and adjusting the parameters ϕ_i's in $\hat{Y}_t = Y_{t-1} + \phi_1 \hat{Y}_{t-1} + \ldots + \phi_{p-1} \hat{Y}_{t-p+1} + \phi_p \hat{Y}_{t-p}$.

Since training is a crucial step for artificial neural networks, the neural network systems with various configurations from 2 nodes to over 40 nodes have been trained by the observations of the commodity volumes in the first twenty years. This allows sufficient time for a neural network to adjust itself to make accurate forecasts. The systems then were tested on forecasting last one to 24 months of observations. The following diagram shows the typical forecasting errors by the recurrent neural networks of different nodes:

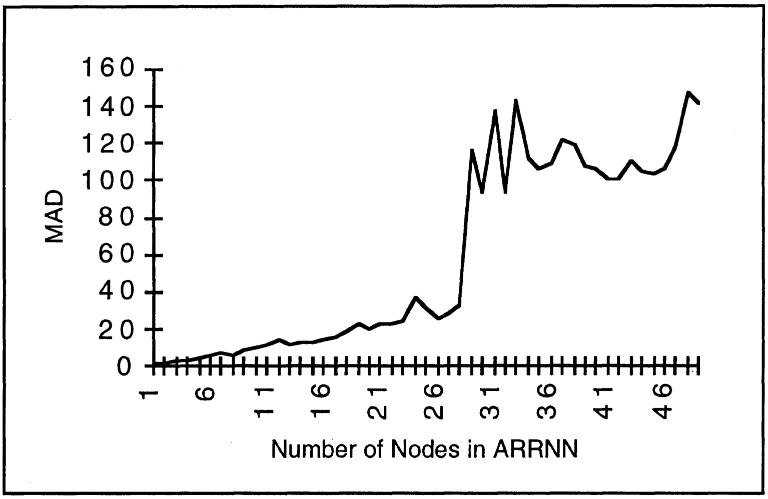


Fig. 2. The MAD of ARRNNs with Various Number of Nodes

In **Fig. 2**, we can easily see that the recurrent neural networks of more nodes tend to have more forecasting errors. This is because the networks of more nodes are more suitable to make forecasts of longer cycles. Nonetheless, when a recurrent neural network has the same number of nodes as the cycle of the data, the forecasting errors tend to be reduced. Hence, when the forecasting error of the neural network of more nodes is not significantly bigger than that of the neural network of less nodes, there is a cyclic trend in the data and the number of nodes in the network indicates its cycle. The results verified that the data is indeed cyclic and it is likely a six-month or twelve-month cycle.

More research is underway to find whether the general recurrent neural networks could be used to assist the general Box-Jenkins ARIMA(p,d,q) models in determining its parameters. A recurrent neural network is being designed and implemented to emulate the general Box-Jenkins ARIMA(p,d,q) model. The weight adjusting strategies of the neural network are being developed to estimate the parameters of the ARIMA(p,d,q) model, to monitor the performance of the forecasting system with these parameters and to modify the parameters when necessary. It would also be interesting to extend the current work to find the similarities and equivalence of other statistical models and artificial neural networks. The ultimate goal of this research work is to improve the forecasting systems in general and automate the parameter estimation of statistical models if such similarities and equivalence could be found. On this basis, we may also find ways to improve the forecasting performance of artificial neural networks.

References

1. Box, G.E.P., G.M. Jenkins and G.C. Reinsel, Time Series Analysis: Forecasting and Control, 3rd ed., Prentice Hall, Englewood Cliffs, New Jersey (1994)
2. Trippi, R.R. and E. Turban, Neural Networks in Finance and Investing, Probus Publishing Company, Chicago, Illinois (1993)
3. White, H., Artificial Neural Networks: Approximation and Learning Theory, Blackwell Publishers, Cambridge, Massachusetts (1992)
4. Ibbotson, R.G. and R.A. Sinquefield, Stocks, Bonds, Bills, and Inflation: Historical Returns (1925-1987), The Institute of Chartered Financial Analysis, Chicago, Illinois (1989)
5. Wu, S.I., An Artificial Neural Network for Stock-Market Forecasting, Proceedings of the International Joint Conference on Neural Networks, Beijing, PROC (1992) I316-321
6. Wu, S.I., Artificial Neural Networks in Forecasting, Neural Networks World, Vol. 4, NO. 2, IDG VSP, Prague, Czechoslovakia (1995) 199-220
7. Wu, S.I., A Comparison of Recurrent Neural Network and Simple Exponential Smoothing in Forecasting, Proceedings of the International Symposium on Artificial Neural Networks, Tainan, Taiwan, ROC (1994) 535-540
8. Wu, S.H., Time Series Forecasts for Commodity Volumes of Port of Kaohsiung, Proceedings of 1993 Joint Conference, Taipei, Taiwan, ROC (1993) 161
9. Shim, I.K. and J.G. Siegel, Handbook of Financial Analysis, Forecasting & Modeling, Prentice Hall, Englewood Cliffs, New Jersey (1988)
10. Bowerman, B.L. and R.T. O'Connell, Time Series Forecasting: Unified Concepts and Computer Implementation, 2nd ed., Duxbury Press, Boston, Massachusetts (1987)
11. Mahoney, D.W., R.P. Lu and S.I. Wu, Construxtion of An Artificial Neural Network for Simple Exponential Smoothing in Forecasting, Proceedings of 1994 Symposium in Applied Computing, Phoenix, Arizona, (1994) 308-312

New Algorithms to Predict Secondary Structures of RNA Macromolecules

Mourad Elloumi

Computer Science Department,
Faculty of Sciences of Tunis, and
Faculty of Economic Sciences and Management of Tunis
El Manar 2092 Tunis, Tunisia.

Abstract. In this paper, we present, under the *Hypothesis of Linearity of Energy* (HLE), new algorithms to solve the problem of the *prediction by energy computation* of RNA stable secondary structures. We present our dynamic programming algorithm to compute the free energies of the stable secondary structures, and our traceback algorithm to *predict* these structures. Our algorithm for computing the free energies is of complexities $O(n^3)$ in computing time and $O(n^2)$ in memory space, where n is the length of the string. Our prediction algorithm is of complexity $O(n*log_2(n))$ in computing time. Compared to other algorithms, under the HLE, our algorithms present the advantage of taking into account the energetic contribution of all the unpaired bases, in addition to the energetic contribution of the paired ones.

0 Introduction

A *RiboNucleic Acid* (RNA) macromolecule can be coded by a string. Each character in this string codes a component of this macromolecule. These components, called *ribonucleotides* or *bases*, are *Adenine*, *Cytosine*, *Guanine* and *Uracil*. They are coded respectively by the characters *A*, *C*, *G* and *U*. Under some thermodynamic conditions, some regions of the macromolecule interact with each other and creat, hence, folds within the macromolecule. These interactions are represented at the level of the primary structure by pairings between the different substrings coding the regions entering in interactions. The primary structure of the macromolecule provided with these pairings is called *secondary structure*. It is easy then to imagine that a macromolecule, represented by its primary structure, can have many secondary structures. However, only one of these structures is *stable* : it is the one that has the minimum *free energy*. The knowledge of this structure plays an important role, not only, to determine the interactions of the macromolecule with the *DesoxyriboNucleic Acid* (DNA) and the proteins, but also, to know its functions and its *biochemical* activities.

The algorithms that predict secondary structures of RNA macromolecules are :

(*i*) Either algorithms that adopt a *regions* approach : we establish the list of all the substrings (*regions*) that can be paired with each other, while respecting the thermodynamic laws. Then, from the different combinations of the unoverlapped pairings, we establish the list of all the possible secondary structures. For each secondary structure, we compute its free energy by using the experimental results [10, 11, 14, 20, 24, 25]. The structure that has the minimum free energy is the *true* secondary structure of the macromolecule.

The algorithms that adopt this approach are, unfortunately, costly in computing time. Among these algorithms, we mention the one of Pipas and Mc Mahon [19], the one of Studnicka *et al.* [23], the one of Martinez [13] and the one of Dumas and Ninio [7]. The algorithm of Pipas and Mc Mahon is the first algorithm to be used to predict secondary structures of RNA macromolecules. Its computing time complexity is of the order of $O(2^n)$, where n is the size of the string [21].

(*ii*) Or algorithms that adopt a *dynamic programming* approach [3, 4]. These algorithms have been developed either under the *Hypothesis of Linearity of Energy* (HLE) or under the *Hypothesis of Loops Dependent Energy* (HLDE) [22, 28]. By using these algorithms, we proceed in two steps :

During the first step, we compute the energy of the stable secondary structure associated with the concerned string (primary structure) : the computation of the energies of the stable secondary structures associated with longer substrings is made by using the computations results of the energies of the stable secondary structures associated with shorter substrings. We reiterate this process until the energy of the stable secondary structure associated with the whole string is computed.

During the second step, we *predict* the pairings that generate the stable secondary structure associated with the concerned string : the prediction of the pairings that generate the stable secondary structures associated with shorter substrings is made according to the pairings that generate the stable secondary structures associated with longer substrings. We reiterate this process until the prediction of the stable secondary structure associated with the whole string is ended.

The algorithms that adopt this approach are less costly. Among these algorithms, we mention the one of Waterman and Smith [27], the one of Nussinov and Jacobson [15], the one of Auron *et al.* [2] and those of Sankoff *et al.* [22]. The order of computing time complexity of these algorithms varies between $O(n^3)$ and $O(n^4)$, where n is the size of the string.

In this paper, we present, under the HLE, our dynamic programming algorithm to compute the free energies of the stable secondary structures, and our traceback algorithm to predict these structures. Compared to other algorithms under the HLE, our algorithms present the avantage of taking into account the energetic contribution of all the unpaired bases, in addition to the energetic contribution of the paired ones.

In the first section of this paper, we present, on one hand, a formal definition of a *secondary structure* and of its different kinds of *loops*, on the other hand, we define the *free energy* and the *loop energy* associated with a substring.

In the second section, we show how we represent a secondary structure and its different loops.

In the third section, we present the different equations of energies computation.

In the fourth section, we present, under the HLE, our dynamic programming algorithm to compute the free energies of the stable secondary structures and our traceback algorithm to predict these structures.

Finally, in the last section, we present our conclusion.

1 Definitions and Notations

A *string* over an alphabet $\mathcal{A}$ is a concatenation of elements of $\mathcal{A}$. The *length* of a string w, denoted by $|w|$, is the number of the characters that constitute this string. By convention, the null length string will be denoted by ε. A string w of length n will be denoted by $w_{1,n}$ and the i^{th} character of w, $1\leqslant i\leqslant n$, will be denoted by w^i. A portion of w that begins at the position i and ends at the position j, $1\leqslant i\leqslant j\leqslant n$, is called *substring* of w and will be denoted by $w_{i,j}$. The *primary structure* of an RNA macromolecule is a string over the alphabet $\mathcal{A}_{RNA}=\{A, C, G, U\}$.

Let w be a primary structure of an RNA macromolecule, the set $\{w^i, w^{i+1}, \ldots, w^j\}$, $0<i\leqslant j\leqslant |w|$, of the characters making up a substring $w_{i,j}$ of w will be denoted by $\mathcal{C}(w_{i,j})$. We define on $\mathcal{C}(w)$ a *pairing* relation, denoted by $\leftrightarrow$, verifying the following properties :

(i) If $w^i\leftrightarrow w^j$ then $(j-i)\geqslant 4$.

(ii) If $w^i\leftrightarrow w^j$ then $w^i=A$ and $w^j=U$, or $w^i=U$ and $w^j=A$, or $w^i=C$ and $w^j=G$, or $w^i=G$ and $w^j=C$, or $w^i=G$ and $w^j=U$, or $w^i=U$ and $w^j=G$. The pair $\{w^i,w^j\}$ is called *Watson-Crick Pair* (WCP).

(iii) If $w^i\leftrightarrow w^j$ then for any k, $k\in[1..i\text{-}1]\cup[i+1..j\text{-}1]\cup[j+1..|w|]$, we can have neither $w^i\leftrightarrow w^k$ nor $w^j\leftrightarrow w^k$.

(iv) For any couples (i,i') and (j,j'), $i'\in]i..j[$ and $j'\in[1..i[\cup]j..|w|]$, if we have $w^i\leftrightarrow w^j$ then we cannot have $w^{i'}\leftrightarrow w^{j'}$.

A *secondary structure* associated with a primary structure w and a pairing relation $\leftrightarrow$, defined on $\mathcal{C}(w)$, is the set $S(w,\leftrightarrow)=\{(w^i,w^j)\mid w^i\leftrightarrow w^j \text{ and } 0<i<j\leqslant |w|\}$. The empty secondary structure will be denoted by ω. A subset $S(w_{i,j},\leftrightarrow)$, $0<i<j\leqslant|w|$, of $S(w,\leftrightarrow)$ such that $S(w_{i,j},\leftrightarrow)=\{(w^p,w^q)\mid w^p\leftrightarrow w^q \text{ and } 0<i\leqslant p<q\leqslant j\leqslant|w|\}$ is called *substructure* of $S(w,\leftrightarrow)$.

With each secondary structure $S(w,\leftrightarrow)$ we associate a negative weight, denoted by $E(w,\leftrightarrow)$, called *free energy* of the structure $S(w,\leftrightarrow)$. The function E is called *energetic function*. The secondary structure for which this energy is minimum is called *stable* secondary structure of the macromolecule. It will be denoted by $S_{min}(w)$ and its free energy will be denoted by $E_{min}(w)$:

$$E_{min}(w)=\begin{cases} min_{\leftrightarrow}\{E(w,\leftrightarrow)\} & \text{if } \exists \leftrightarrow \text{ on } \mathcal{C}(w) \\ E''(w) & \text{else} \end{cases} \tag{1}$$

The function E'' is an energetic function dependent solely on the nature of the bases which constitute the string w. By convention, we will set $E''(\varepsilon)=0$.

Let us consider now a substring $w_{i,j}$, $0<i<j\leqslant|w|$, the *loop energy*, denoted by $E_{loop}(w_{i,j})$, associated with the substring $w_{i,j}$ is the minimum free energy that can have a secondary structure of $w_{i,j}$ containing the couple (w^i,w^j) :

$$E_{loop}(w_{i,j})=\begin{cases} min_{\leftrightarrow\,\mid w^i\leftrightarrow w^j}\{E(w_{i,j},\leftrightarrow)\} & \text{if} \exists \leftrightarrow \text{ on } \mathcal{C}(w_{i,j})\mid w^i\leftrightarrow w^j \\ +\infty & \text{else} \end{cases} \tag{2}$$

Each secondary structure $S(w,\leftrightarrow)$ can be subdivided in a unique way in a certain number of *loops*. We distinguish five types of loops :

(i) If $w^i \leftrightarrow w^j$ and the bases $w^{i+1}, w^{i+2}, \ldots , w^{j-1}$ are not paired then the singleton $\eta_{i,j}(w)=\{(w^i,w^j)\}$ is called *hairpin loop.*

(ii) If $w^i \leftrightarrow w^j, w^{i+1} \leftrightarrow w^{j-1}, \ldots , w^{i+k} \leftrightarrow w^{j-k}$, with $k \geq 1$, then the set $\sigma^k{}_{i,j}(w)=\{(w^i, w^j), (w^{i+1}, w^{j-1}), \ldots , (w^{i+k},w^{j-k})\}$ is called *stack.*

(iii) If $w^i \leftrightarrow w^j$ and $w^{i+k} \leftrightarrow w^{j-1}$ (resp. $w^i \leftrightarrow w^j$ and $w^{i+1} \leftrightarrow w^{j-k}$), with $i+1<i+k<j-1$ (resp. $i+1<j-k<j-1$), and the bases $w^{i+1}, w^{i+2}, \ldots , w^{i+k-1}$ (resp. $w^{j-k+1}, w^{j-k+2}, \ldots , w^{j-1}$) are not paired then the pair $\lambda^k{}_{i,j}(w)=\{(w^i,w^j), (w^{i+k},w^{j-1})\}$ (resp. $\rho^k{}_{i,j}(w)=\{(w^i,w^j), (w^{i+1},w^{j-k})\}$) is called *left bulge loop* (resp. *right bulge loop*)

(iv) If $w^i \leftrightarrow w^j$ and $w^{i+\ell} \leftrightarrow w^{j-m}$, with $i+1<i+\ell<j-m<j-1$, and the bases $w^{i+1}, w^{i+2}, \ldots , w^{i+\ell-1}$ and $w^{j-m+1}, w^{j-m+2}, \ldots , w^{j-1}$ are not paired then the pair $\zeta^{\ell,m}{}_{i,j}(w)=\{(w^i,w^j), (w^{i+\ell},w^{j-m})\}$ is called *interior loop.*

(v) If $w^i \leftrightarrow w^j, w^{i+k_1} \leftrightarrow w^{i+\ell_1}, w^{i+k_2} \leftrightarrow w^{i+\ell_2}, \ldots , w^{i+k_m} \leftrightarrow w^{i+\ell_m}$, with $i<i+k_1<i+\ell_1<i+k_2<i+\ell_2<\ldots<i+k_m<i+\ell_m<j$, and for any k, $k \in \;]i..i+k_1[\cup]i+\ell_1..i+k_2[\cup \ldots \cup]i+\ell_m..j[$, we have w^k is not paired then the set $\mu^{k_1,\ell_1,\ldots,k_m,\ell_m}{}_{i,j}(w)=\{(w^i,w^j), (w^{i+k_1},w^{i+\ell_1}), (w^{i+k_2},w^{i+\ell_2}), \ldots , (w^{i+k_m},w^{i+\ell_m})\}$ is called *multiloop.*

The set of loops that constitute a secondary structure $S(w,\leftrightarrow)$ will be denoted by $\mathscr{L}(w,\leftrightarrow)$ and the set of loops that constitute the stable secondary structure $S_{min}(w)$ will be denoted by $\mathscr{L}_{min}(w)$.

Let ℓ_i be one of the loops of a secondary structure $S(w,\leftrightarrow)$. An unpaired base w^k of w is said to be *accessible* from ℓ_i, if and only if, there is a couple (w^p,w^q) of ℓ_i such that $p<k<q$ and there is no other couple (w^ℓ,w^m) belonging to $S(w,\leftrightarrow)$ but not belonging to ℓ_i such that $p<\ell<k<m<q$.

The set of non accessible bases from any loop of $\mathscr{L}(w,\leftrightarrow)$ is called the *tail* of the secondary structure $S(w,\leftrightarrow)$ and will be denoted by $\tau(w,\leftrightarrow)$. The tail of the stable secondary structure $S_{min}(w)$ will be denoted by $\boldsymbol{\tau}_{min}(w)$.

2 Representation of a Secondary Structure and Its Loops

A secondary structure $S(w,\leftrightarrow)$ can be represented by an undirected graph G, $G=(V,E)$, such that $V=\mathscr{C}(w)$ and $E=S(w,\leftrightarrow)$. A loop ℓ of $S(w,\leftrightarrow)$ defined on $\mathscr{C}(w_{i,j})$, $0<i<j\leq|w|$, can be represented by a subgraph G', $G'=(V',E')$, of G such that $V'=\mathscr{C}(w_{i,j})$ and $E'=\ell$. When the edges of G are represented by segments with equal lengths, we say that we have a *normal* representation of the structure $S(w,\leftrightarrow)$.

Fig. 1 is a normal representation of a secondary structure with its different loops.

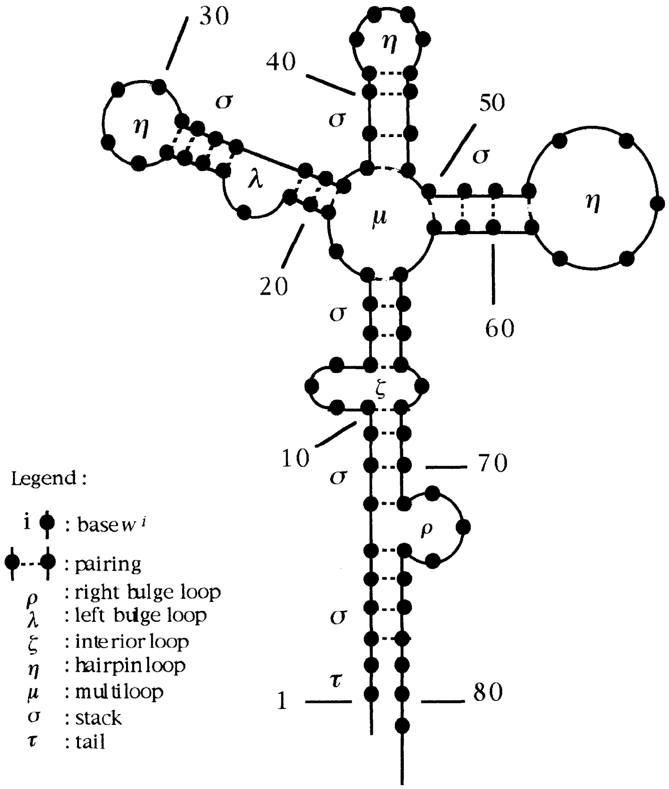


Fig. 1. Normal representation of a secondary structure

3 Equations of Energies Computation

As we have explained in the introduction, to predict the stable secondary structure of an RNA macromolecule, we are brought to compute the minimum free energy that can have a secondary structure of this macromolecule. Unfortunately, the computing methods based on principles of thermodynamic do not permit to compute this energy. Whereas, it is experimentally possible to determine the energetic contribution of a WCP or a loop [10, 11, 14, 20, 24, 25]. A hypothesis introduced by biochemists consists in supposing that the free energy of a secondary structure depends not only on the pairs that constitute this structure, but also, on the other unpaired bases of the macromolecule [28]. We will call this hypothesis, *Hypothesis of Loops Dependent Energy* (HLDE) :

$$E_{min}(w) = \begin{cases} min_{\leftrightarrow}\left\{ \sum_{\ell_i \in \mathcal{L}(w,\leftrightarrow)} E'(\ell_j) + \sum_{w^r \in \mathcal{T}(w,\leftrightarrow)} e'(w^r) \right\} & \text{if } \exists \leftrightarrow \text{on } \mathcal{C}(w) \\ E''(w) = \sum_{s=1}^{|w|} e'(w^s) & \text{else} \end{cases} \quad (3)$$

The function E' is an energetic function that depends on both the pairs that constitute the loop ℓ_i and the accessible bases from this loop. The values of the energetic function E' are negative for the stacks and positive for the other loops [10, 11, 14, 20, 24, 25]. The function e' is a positive energetic function dependent only on the nature of the base in question.

In the particular case where for any loop ℓ_i, $\ell_i \in \mathscr{L}(w,\leftrightarrow)$, we have :

$$E'(\ell_i) = \sum_{\substack{(w^p, w^q) \in \ell_i \\ \text{and } (w^p, w^q) \notin \ell_h,\ (h \lhd i)}} e(w^p, w^q) + \sum_{\substack{w^s \text{accessible} \\ \text{from } \ell_i}} e'(w^s) \tag{4}$$

we say that the function E is *linear*. We will call this hypothesis, *Hypothesis of Linearity of Energy* (HLE) [22].

Our dynamic programming algorithm to compute the free energies of the stable secondary structures, and our traceback algorithm to predict these structures, under the HLE, use the following theorem :

Lemma 1. For any substring $w_{i,j}$, $0<i\leqslant j-4\leqslant |w|$, of a primary structure w and for any k, $i<k\leqslant j$, if $(w^i,w^k)\in S_{min}(w_{i,j})$ then under the HLE we have :

$$E_{min}(w_{i,j})=E_{min}(w_{i+1,k-1})+e(w^i,w^k)+E_{min}(w_{k+1,j}).$$

Proof. By definition, we have $E_{min}(w_{i,j})=min_{\leftrightarrow}\{E(w_{i,j},\leftrightarrow)\}$. Since the couple $(w^i,w^k)\in S_{min}(w_{i,j})$, we eliminate then all the substructures that do not contain this couple. Hence, $E_{min}(w_{i,j})=min_{\leftrightarrow \text{ and } w^i\leftrightarrow w^k}\{E(w_{i,j},\leftrightarrow)\}$. Since the concerned substructures are those that contain the couple (w^i,w^k), then thanks to the HLE we have : $E_{min}(w_{i,j})=min_{\leftrightarrow}\{E(w_{i+1,k-1},\leftrightarrow)+e(w^i,w^k)+E(w_{k+1,j},\leftrightarrow)\}$. Since the function e is independante of the pairings, we have then : $E_{min}(w_{i,j})=min_{\leftrightarrow}\{E(w_{i+1,k-1},\leftrightarrow)\}+e(w^i,w^k)+min_{\leftrightarrow}\{E(w_{k+1,j},\leftrightarrow)\}$. Hence, by Equation (1) we have :

$$E_{min}(w_{i,j})=E_{min}(w_{i+1,k-1})+e(w^i,w^k)+E_{min}(w_{k+1,j}) \qquad \bullet$$

Theorem 1. For any substring $w_{i,j}$, $0<i\leqslant j-4\leqslant |w|$, of a primary structure w, we have under the HLE :

$$E_{min}(w_{i,j})=\begin{cases} min\Big\{e'(w^i)+E_{min}(w_{i+1,j}), \\ \quad \displaystyle\min_{\substack{i+4\leqslant k\leqslant j \\ \text{and} \\ w^i\leftrightarrow w^k}} \{E_{min}(w_{i+1,k-1})+e(w^i,w^k)+E_{min}(w_{k+1,j})\}\Big\} & \text{if } \exists \leftrightarrow \text{ on } \mathscr{E}(w_{i,j}) \\ \displaystyle\sum_{s=i}^{j} e'(w^s) & \text{else} \end{cases}$$

Proof. If there exist pairings on $\mathcal{C}(w_{i,j})$ then the secondary structures of $w_{i,j}$ are in one of the following cases :

(a) either there is no base w^k, $i<k\leq j$, such that (w^i,w^k) belongs to this structure,

(b) or there is a base w^k, $i<k\leq j$, such that (w^i,w^k) belongs to this structure.

If the stable secondary structure $S_{min}(w_{i,j})$ is in the case (a) then we have obviously $S_{min}(w_{i,j})=S_{min}(w_{i+1,j})$. Therefore we have too $\mathcal{L}_{min}(w_{i,j})=\mathcal{L}_{min}(w_{i+1,j})$. On the other hand, under the HLDE, therefore under the HLE too, we have :

$$E_{min}(w_{i,j}) = \sum_{\ell_k \in \mathcal{L}_{min}(w_{ij})} E'(\ell_k) + \sum_{w^r \in \mathcal{T}_{min}(w_{ij})} e'(w^r)$$

The base w^i is not paired then it belongs to $\mathcal{T}_{min}(w_{i,j})$. We can then rewrite $E_{min}(w_{i,j})$ as follows :

$$E_{min}(w_{i,j}) = \sum_{\ell_k \in \mathcal{L}_{min}(w_{ij})} E'(\ell_k) + \sum_{\substack{w^r \in \mathcal{T}_{min}(w_{ij}) \\ \text{and} \\ w^r \neq w^i}} e'(w^r)+e'(w^i).$$

Or, thanks to the equality between $\mathcal{L}_{min}(w_{i,j})$ and $\mathcal{L}_{min}(w_{i+1,j})$:

$$E_{min}(w_{i,j}) = \sum_{\ell_k \in \mathcal{L}_{min}(w_{i+1\,j})} E'(\ell_k) + \sum_{w^r \in \mathcal{T}_{min}(w_{i+1,j})} e'(w^r)+e'(w^i)\cdot$$

Hence, thanks to Equation (3) :

$$E_{min}(w_{i,j})=E_{min}(w_{i+1,j})+e'(w^i).$$

On the other hand, if the stable secondary structure is in the case (b) then let us call k_0 the position in $w_{i,j}$ such that $(w^i,w^{k_0})\in S_{min}(w_{i,j})$. From Lemma 1, we have :

$$E_{min}(w_{i,j})=E_{min}(w_{i+1,k_0-1})+e(w^i,w^{k_0})+E_{min}(w_{k_0+1,j}).$$

Since $E_{min}(w_{i,j})$ is minimum, we have then :

$$E_{min}(w_{i,j})=min_{i+4\leq k\leq j}\{E_{min}(w_{i+1,k-1})+e(w^i,w^k)+E_{min}(w_{k+1,j})\}.$$

By considering the cases (a) and (b) together, since we seek to minimize the value of the energy $E_{min}(w_{i,j})$, we have then :

$$E_{min}(w_{i,j})=min\{e'(w^i)+E_{min}(w_{i+1,j}),$$
$$min_{i+4\leq k\leq j}\{E_{min}(w_{i+1,k-1})+e(w^i,w^k)+E_{min}(w_{k+1,j})\}\}.$$

Finally, according to Equation (3), if there are no pairings on $\mathcal{C}(w_{i,j})$ then we have :

$$E_{min}(w_{i,j})=\sum_{s=i}^{j} e'(w^s)$$

•

We present now, under the HLE, our dynamic programming algorithm, guided by a base-to-base pairing, that computes the free energy of the stable secondary structure, then, we present our algorithm that predicts this structure by basing itself on tracing back the matrix filled by the previous algorithm. In [8], we have also presented, under the HLDE, our other dynamic programming algorithm that computes the free energy of the stable secondary structure, and our other traceback algorithm that predicts this structure.

4 Algorithms of Energies Computation and Prediction

The algorithm that computes the free energies $E_{min}(w_{i,j})$, $0<i\leqslant j\leqslant |w|$, under the HLE is the following :

Algorithm 1.

```
(i) (i.a) Construct a matrix $M$ of size $(|w|*|w|)$;
    (i.b) for any $(i,j)$, $0\leqslant j-i<4$, do
            $s:=0$;
            for $k:=i$ to $j$ do $s:=s+e'(w^k)$ endfor;
            $M[i,j]:=s$
          endfor;
(ii) for $j:=5$ to $|w|$ do
       for $i:=j-4$ downto 1 do
         (ii.a) $m_1:=e'(w^i)+M[i+1,j]$ and $m_2:=+\infty$ ;
         (ii.b) for $k:=i+4$ to $j$ do
                  if $\{w^i,w^k\}$ is a WCP then
                    $m_2:=min\{m_2, M[i+1,k-1]+e(w^i,w^k)+M[k+1,j]\}$
                  endif
                endfor;
         (ii.c) $M[i,j]:=min\{m_1, m_2\}$
       endfor
     endfor.
```

Proposition 1. Let w be the primary structure of an RNA macromolecule. For any couple (i,j), $0<i\leqslant j\leqslant |w|$, Algorithm 1 computes the energy $E_{min}(w_{i,j})$ under the HLE and we have $M[i,j]=E_{min}(w_{i,j})$.

Proof. For any couple (i,j), $0<i\leqslant j\leqslant |w|$, two cases are possible : (a) either $(j-i)<4$, (b) or $(j-i)\geqslant 4$.

If we are in the case (a) then there do not exist pairings on $\mathcal{C}(w_{i,j})$. Thanks to the step (i), we have :

$$M[i,j]=\sum_{s=i}^{j} e'(w^s)$$

By Theorem 1, we have :

$$M[i,j]=E_{min}(w_{i,j}).$$

Let us place ourselves now in the case (b). Thanks to the steps ($ii.a$) and ($ii.b$), we have : $m_1:=e'(w^i)+M[i+1,j]$ and

$$\begin{cases} m_2:=min_{i+4\leqslant k\leqslant j}\{M[i+1,k-1]+e(w^i,w^k)+M[k+1,j]\} & \text{if } \exists\, k, i+4\leqslant k\leqslant j, \\ & \text{such that } \{w^i,w^k\} \text{ is a WCP} \\ m_2:=+\infty & \text{else} \end{cases}$$

When there exists k, $i+4\leqslant k\leqslant j$, such that $\{w^i, w^k\}$ is a WCP then thanks to the step ($ii.c$), we have:

$$M[i,j]=min\{m_1, m_2\}.$$

According to Theorem 1, we have :

$$M[i,j]=E_{min}(w_{i,j}).$$

When there exists no k, $i+4\leqslant k\leqslant j$, such that $\{w^i, w^k\}$ is a WCP, we have obviously, thanks to the step (*ii.c*) :

$$M[i,j]=e'(w^i)+M[i+1,j].$$

If in addition there are no $k_1, k_2, \ldots, k_{j-i-4}$ ($i+5\leqslant k_1\leqslant j$, $i+6\leqslant k_2\leqslant j$, ... , j-$4\leqslant k_{j-i-4}\leqslant j$) such that $\{w^{i+1}, w^{k_1}\}$, $\{w^{i+2}, w^{k_2}\}$, ... , $\{w^{j-4}, w^{k_{j-i-4}}\}$ are WCPs, i.e., there are no pairings on $\mathcal{C}(w_{i,j})$, then we have :

$$M[i,j]=e'(w^i)+M[i+1,j]=\sum_{s=i}^{i+1} e'(w^s)+M[i+2,j]=\ldots=\sum_{s=i}^{j-5} e'(w^s)+M[j-4,j].$$

Since for $(j-4,j)$ we have :

$$m_1=e'(w^{j-4})+M[j-3,j]=e'(w^{j-4})+\sum_{s=j-3}^{j} e'(w^s)$$

(thanks to the step (*i*), since $j-(j-3)=3<4$) and $m_2=+\infty$ (thanks to the step (*ii*), since there does not exist k_{j-i-4}, j-$4\leqslant k_{j-i-4}\leqslant j$, such that $\{w^{j-4}, w^{k_{j-i-4}}\}$ is a WCP), then thanks to the step (*ii.c*) we have :

$$M[j-4,j]=m_1=e'(w^{j-4})+M[j-3,j]=e'(w^{j-4})+\sum_{s=j-3}^{j} e'(w^s)=\sum_{s=j-4}^{j} e'(w^s).$$

Hence, we have :

$$M[i,j]=\sum_{s=i}^{j-5} e'(w^s)+\sum_{s=j-4}^{j} e'(w^s)=\sum_{s=i}^{j} e'(w^s).$$

From Theorem 1, $M[i,j]=E_{min}(w_{i,j})$. •

Proposition 2. Algorithm 1 is of complexities $O(|w|^3)$ in computing time and $O(|w|^2)$ in memory space.

Proof. For any couple (i,j), $0<i<j\leqslant|w|$, we look for the position k_0, $i+4\leqslant k_0\leqslant j$, such that $\{w^i, w^{k_0}\}$ is a WCP and $M[i+1,k_0-1] + e(w^i,w^{k_0}) + M[k_0+1,j] = min_{i+4\leqslant k\leqslant j}\{M[i+1,k-1] + e(w^i,w^k) + M[k+1,j]\}$. This search is made linearly by incrementing the position k, $i+4\leqslant k\leqslant j$. Therefore for a couple (i,j), this search is of complexity $O(|w|)$ in computing time. We have $(|w|-4)^2/2$ couples (i,j) to process (since we must have $(j-i)\geqslant 4$). Therefore, Algorithm 1 is of complexity $O(|w|^3)$ in computing time.

Algorithm 1 uses a memory space equal to $|w|^2/2$, then it is of complexity $O(|w|^2)$ in memory space. •

Let us present now our prediction algorithm under the HLE. This algorithm traces back the matrix M filled thanks to Algorithm 1. The list L represents the set of couples making up the stable secondary structure of $w_{i,j}$. It is initialized to the empty list at the first call to Algorithm 2.

```
Algorithm 2 $(i,j)$
  $(i)$ if $(j\text{-}i)\geq 4$ then
    $(i.a)$ if $M[i,j]=e'(w^i)+M[i+1,j]$ then Algorithm 2 $(i+1,j)$
            else
    $(i.b)$ $k:=i+4$;
            while $(M[i,j]\neq M[i+1,k\text{-}1]+e(w^i,w^k)+M[k+1,j])$ do
              $k:=k+1$
            endwhile;
    $(i.c)$ $L:=L\cup\{(w^i,w^k)\}$;
    $(i.d)$ Algorithm 2 $(i+1,k\text{-}1)$; Algorithm 2 $(k+1,j)$;
            endif;
  endif;
  $(ii)$ $S_{min}(w_{i,j}):=L$.
```

Proposition 3. Let $w_{i,j}$ be a substring of the primary structure w and let M be the matrix filled thanks to Algorithm 1. Algorithm 2 gives, under the HLE, the substructure $S_{min}(w_{i,j})$.

Proof. By Proposition 1, we have $M[i,j]=E_{min}(w_{i,j})$ and by Theorem 1, if there exists a pairing on $\mathscr{C}(w_{i,j})$ then two cases are possible : (a) either $M[i,j]=e'(w^i)+M[i+1,j]$, (b) or $M[i,j]=min_{i+4\leq k\leq j}\{M[i+1,k\text{-}1]+e(w^i,w^k)+M[k+1,j]\}$.

If we are in the case (a) then we have :

$$S_{min}(w_{i,j})=S_{min}(w_{i+1,j}).$$

The step $(i.a)$ processes this case and computes $S_{min}(w_{i+1,j})$ by a recursive call to Algorithm 2.

On the other hand, if we are in the case (b) then let k_0 be the position in $w_{i,j}$ such that $M[i,j]=M[i+1,k_0\text{-}1]+e(w^i,w^{k_0})+M[k_0+1,j]=min_{i+4\leq k\leq j}\{M[i+1,k\text{-}1]+e(w^i,w^k)+M[k+1,j]\}$, i.e., such that $(w^i,w^{k_0})\in S_{min}(w_{i,j})$. The loop of the step $(i.b)$ finds this k_0 and the step $(i.c)$ inserts the couple (w^i,w^{k_0}) in the liste L. Since $(w^i,w^{k_0})\in S_{min}(w_{i,j})$, then from Lemma 1, we have :

$$S_{min}(w_{i,j})=S_{min}(w_{i+1,k_0\text{-}1})\cup\{(w^i,w^{k_0})\}\cup S_{min}(w_{k_0+1,j}).$$

The step $(i.d)$ computes both of the substructures $S_{min}(w_{i+1,k_0\text{-}1})$ and $S_{min}(w_{k_0+1,j})$ thanks to two recursive calls to Algorithm 2.

When $(j\text{-}i)<4$, it is obvious that there do not exist pairings on $\mathscr{C}(w_{i,j})$, and consequently, the list L remains unchanged.

Finally, thanks to the step (ii), when the list L contains all the couples making up the sought-after secondary structure, we assign it to $S_{min}(w_{i,j})$.

Proposition 4. Algorithm 2 is of complexity $O(|w|*log_2(|w|))$ in computing time.

Proof. Each call to Algorithm 2 generates, at most, two other recursive calls to Algorithm 2. Therefore for a primary structure w we have, at most, $log_2(|w|)$ recursive levels. During a call concerning a couple (i,j), $0<i\leq j\text{-}4\leq|w|$, we look for the position

k_0, $i+4 \leqslant k_0 \leqslant j$, such that $\{w^i, w^{k_0}\}$ is a WCP and $M[i+1,k_0-1]+e(w^i,w^{k_0}) +M[k_0+1,j]=min_{i+4\leqslant k\leqslant j}\{M[i+1,k-1]+e(w^i,w^k)+M[k+1,j]\}$. This search is of complexity $O(|w|)$ in computing time. Therefore Algorithm 2 is of complexity $O(|w|*log_2(|w|))$ in computing time.

5 Conclusion

In this paper, we have presented new algorithms to solve the problem of the *prediction by energy computation* of the stable secondary structures of RNA macromolecules, under the *Hypothesis of Linearity of Energy* (HLE). Our algorithms are more efficient than the other algorithms, under the HLE. Indeed, our algorithms take into account the energetic contribution of all the unpaired bases, in addition to the energetic contribution of the paired ones. The other algorithms often ignore the energetic contribution of some unpaired bases : the algorithms of Nussinov and Jacobson [15] ignore the energetic contribution of all the unpaired bases, the ones of Waterman and Smith [27] ignore as well the energetic contribution of the multiloops as the one of the tails, and the ones of Sankoff [22] ignore the energetic contribution of the tails.

On the other hand, our algorithms solve the problem of the prediction by energy computation, under the HLE, within a time of the order of $O(n^3)$ and using a memory space of the order of $O(n^2)$. These complexities are known to be the best existing complexities to solve this problem under the HLE.

References

1. Aho, A. V., Hopcroft, J. E., Ullman, J. D.: The Design and Analysis of Computer Algorithms. Addison-Wesley Publishing Company) (1974) 60-65

2. Auron, B. E., Rindowe, W. P., Vary, C. P. H., Celentano, J. J., Vournakis, J. N.: Computer Aided Prediction of RNA secondary structures. Nucleic Acids Research, Vol. 10, N°1 (1982) 403-419

3. Bellman, R. E.: Dynamic Programming. Princeton University Press, New Jersey (1957)

4. Bellman, R. E., Dreyfus, S. E.: Applied Dynamic Programming. Princeton University Press, New Jersey (1962)

5. Berge, C.: Graphes et Hypergraphes. Dunod Editeur, Paris (1970)

6. Delisi, Ch., Crothers, D. M.: Prediction of RNA Secondary Structure. Proc. Nat. Acad . Sci. USA, Vol. 68, N°11 (Nov. 1971) 2682-2685

7. Dumas, J. P., Ninio, J.: Efficient Algorithms for Folding and Comparing Nucleic Acid Sequences. Nucleic Acids Research, Vol. 10, N°1 (1982) 197-206

8. Elloumi, M.: Analyse de Chaînes de Caractères Codant des Macromolécules Biologiques. Science Doctorate Dissertation, The University of Aix-Marseilles III, France (June 1994)

9. Gralla, J., Steitz, J. A., Crothers, D. M.: Direct Physical Evidence for Secondary Structure in an Isolated Fragment of R17 Bacteriophage mRNA. Ibid, N°248 (1974) 204-208

10. Gralla, J., Crothers, D. M.: Free Energy of Imperfect Nucleic Acid Helices II, Small Hairpin Loops. J. Mol. Biol., N°73 (1973) 497-511

11. Gralla, J., Crothers, D. M.: Free Energy of Imperfect Nucleic Acid Helices III, Small Internal Loops Resulting from Mismatches. J. Mol. Biol., N°78 (1973) 301-319

12. Larmore, L. L., Schieber, B.: On-line Dynamic Programming with Applications to the Prediction of RNA Secondary Structure. J. of Algorithms, N°12 (1991) 490-515

13. Martinez, H.: A New Algorithm for Calculating RNA Secondary Structure, Manuscript (1980)

14. Ninio, J.: Prediction of Pairing Schemes in RNA Molecules-Loop Contributions and Energy of Wobble and Non-Wobble Pairs. Biochimie, N°61 (1979) 1133-1150

15. Nussinov, R., Jacobson, A.: Fast Algorithm for Predicting the Secondary Structure of Single-Stranded RNA. Proc. Nat. Acad . Sci. USA, Vol. 77, N°11 (Nov. 1980) 6309-6313

16. Oppenheimer, N. J., James, T. L.: Nuclear Magnetic Resonance : Part A, Spectral Techniques and Dynamics. Methods in Enzymology, Vol. 176 (1989)

17. Oppenheimer, N. J., James, T. L.: Nuclear Magnetic Resonance : Part B, Structure and Mechanism. Methods in Enzymology, Vol. 177 (1989)

18. Papanicolaou, C., Gouy, M., Ninio, J.: An Energy Model that Predicts the Correct Folding of both ^{t}RNA and 5sRNA Molecules. Nucleic Acids Research, Vol. 2, N°31 (1984)

19. Pipas, J. M., Mc Mahon, J. E.: Method for Prediction RNA Secondary Structure. Proc. Nat. Acad . Sci. USA, Vol. 72, N°6 (June 1975) 2017-2021

20. Salser, W.: Globin Messenger-RNA Sequences : Analysis of Base-Pairing and Evolutionary Implications. Cold Spring Harbor Symp. Quant. Biol., N°42 (1977) 985-1002

21. Sankoff, D., Morin, A. M., Cedergren, R. J.: The Evolution of 5sRNA Secondary Structure. Journal Canadien de Biochimie, N°56 (1978) 440-443

22. Sankoff, D., Kruskal, J. B., Mainville, S., Cedergren, R.: Fast Algorithms to Determine RNA Secondary Structures Containing Multipleloops. Time Warps, String Edits and Macromolecules : The Theory and Practice of Sequence Comparison. MA : Addison-Wesley (Eds) (1983) 93-120

23. Studnicka, G. M., Rahn, G. M., Cummings, I. W., Salser, W. A.: Computer Method for Predicting the Secondary Structure of Single-Stranded RNA. Nucleic Acids Research, Vol. 5, N°9 (1978) 3356-3387

24. Tinoco Jr., I., Borer, Ph. N., Dengler, B., Levine, M. D., Uhlenbeck, O. C., Crothers, D. M., Gralla, J.: Improved Estimation of Secondary Structure in Ribonucleic Acids. Nature New Biology, N°246 (1973) 40-41

25. Uhlenbeck, O. C., Borer, Ph. N., Dengler, B., Tinoco, I.: Stability of RNA Hairpin Loops : A_6-C_m-U_6. J. Mol. Biol., N°73 (1973) 483-496

26. Waterman, M. S.: Secondary Structure of Single-Stranded Nucleic Acids. Studies in Foundations and Combinatorics, Advances in Mathematics Supplementary Studies, Vol. 1 (1978) 167-212

27. Waterman, M. S., Smith, T. F.: RNA Secondary Structure : A Complete Mathematical Analysis. Mathematical Biosciences, N°42 (1978) 257-266

28. Zuker, M.: The Use of Dynamic Programming Algorithms in RNA Secondary Structure Prediction. Mathematical Methods for DNA Sequences, CRC Press Inc. (Eds.) (1989) 168-170

29. Zuker, M., Stiegler, P.: Optimal Computer Folding of Large RNA Sequences Using Thermodynamics and Auxiliary Information. Nucleic Acids Research, Vol. 9, N°1 (1981) 133-148

A Hybrid GA Statistical Method for the Forecasting Problem : The Prediction of the River Nile Inflows

Ashraf H. Abdel-Wahab *, Mohammed E. El-Telbany*, and Samir I. Shaheen **

* Computers and Systems Dept.,
Electronics Research Institute, Dokki, Giza, Egypt
E-mail: ashraf@eri.sci.eg, telbany@eri.sci.eg
** Computers Engineering Department, Faculty of Engineering, Cairo University.
E-mail: sshaheen@frcu.eun.eg

Abstract : The prediction of time series phenomena is a hard and complex task. Many statistical models have been used for solving such task. The selection of a proper statistical model and the setup of its parameters (in terms of the number of parameters and their values) are difficult tasks and they are usually solved by trial and error. This paper presents a hybrid system that integrates genetic algorithms -as a search algorithm- and traditional statistical models to overcome the model selection and tuning problems. The system is applied to the domain of river Nile inflows forecasting which is characterized by the availability of large amount of data and prediction models. The model sdeveloped by the proposed system are then compared with other models like traditional statistical methods and ANNs.

1 Introduction

The forecasting of time series phenomena using the available data is a very complex task. Stochastic mathematical models like autoregressive moving average (ARMA) and its periodic form (PARMA), generally, do not capture all the characteristics of the available data. As a result of the complexity of the statistical techniques, inaccurate model identification and, parameter estimation, other artificial intelligence techniques have been tried. For example, artificial neural networks which have been applied to the *white Nile river* forecasting based on a routing model from the Packwatch (Lake Albert exit) to Wadi Halfa (Lake Nasser entrance) [Georgakakos, A., P., and Yao, H., 1994].

In this paper, a hybrid system that integrates both, statistical models and a genetic based search technique is presented and discussed. Genetic algorithms are used to search for the best prediction model and its parameter settings (*off-line* learning mode) [Goldberg, 1989]. Also, they are used to adapt the selected prediction model dynamically while observing new data points (*on-line* learning mode). In the *off-line* mode the system extracts the appropriate model and its parameter values depending on pervious available data, and in the *on-line* mode, it adapts the model and its parameter values according to real-time observations. Among the most used traditional statistical methods one may list; ***least squares*** analysis used to model and investigate the relationship between two or more variables, ***Moving averages*** which is helpful in looking for patterns in data when the underlying signal is sensitive to noise, ***Exponential smoothing methods*** which are probably the most widely used class of procedures for smoothing discrete time series in order to forecast the immediate future and finally, ***The Box-Jenkins methodology*** which consists, on one hand, of a large class of time series models and, on the other, of a set of procedures for choosing models from this class to fit to data from a given series [Box G., E., P., and Jenkins, G., M., 1976].

2 The Proposed Genetic Based Forecasting System GBFS

The implemented Genetic based forecasting system is based upon variable length binary chromosomes with fixed size population. Raw data is first examined and processed to eliminate the existing noise by applying the following equation:

$$\tilde{y}_t = y_t - \mu$$

where μ is the mean value of the samples. It should be noted that missing data points are set to zero value. Each chromosome in the population represents a model and its parameter values (a variable length binary string). Initially, the population is initialized randomly for the parameters numbers *p,q* and their values. The crossover operator is slightly modified to accommodate variable length chromosome such that the crossover point is selected to be within the boundaries of the shortest chromosome (single point crossover is adopted) and to perform cut off without disturbing the parameters values. The mutation operator is used in its conventional form. The genetic algorithm cycle continues until the prediction error produced by the best chromosome is within a certain error limit. Models considered in the system (used to initialize the population) are based upon Box-Jenkins time series ARMA(p,q) models]. The identified models are based on the PARMA time series models as shown in the equation below.

$$y_{i,j} = \sum_{k=1}^{p} \phi_k y_{i,j-k} + \sum_{k=1}^{q} \theta_k e_{i,j-k} + \sum_{L=1}^{P} \Phi_L y_{i-L,j} + \sum_{L=1}^{Q} \Theta_L e_{i-L,j} + e_{i,j}$$

The PARMA(p,q)(P,Q) mathematical model, where $y_{i,j}$ is the stochastic process under forecasting for step t, and t = (i-1)*season+j. The values of $\phi_k, \theta_k, \Phi_L, \Theta_L$ are the nonseasonal and seasonal autoregressive and moving average coefficients respectively, and assume $e_{i,j}$ to be the white noise with mean zero and variance δ. Also the values of p, q, P, and Q represent the orders of nonseasonal and seasonal terms.*

The fitness function used for ranking the chromosome (model) in the population is the *Akaike Information Criteria* (AIC) [Tong, H., 1995] which is calculated as follows:

$$AIC(p,q) = N \ln(\Omega_e^2) + 2(p+q)$$

Where N is the number of trained observations, $(p+q)$ the number of parameters estimated, and the maximum likelihood is determined according to

$$\Omega_e^2 = \frac{1}{N} \sum_{t=0}^{N} \tilde{e}^2(t) \quad .$$

The discovered model is then used to predict the future values for a specified horizon. In the off-line learning mode, the forecasting error is based upon a portion of the data that has not been used for training (unseen samples). The forecasted value produced by the system is then compared with the actual value and the forecast error is then calculated. In the on-line learning mode, the current actual reading (data sample) is used as part of the data set to adjust the model parameters in the next time step.

The system performance was better than the conventional one (in terms of the forecasting error in the off-line mode) but still considerably high. Thus, in order to fine tune the resulting solution, the GBFS system has been augmented by a simulated annealing algorithm.
The operation of the hybrid system may be stated as follows: the genetic algorithms search and find the best model structure and its parameters, and simulated annealing started to ensure its globality. Simulating annealing is performed by randomly *perturbing* the independent variables and keeping track of the best (lowest error) fitness individual for each randomized set of variables. A relatively high standard deviation for the random number generator is used at first. After many trials, the set that produced the best chromosome fitness is designated to be the center about which *perturbation* will take place for the next temperature. The temperature (standard deviation of the random number generator) is then reduced, and new trieals are done. The temperature is lowered in steps using the *exponential reduction* by multiplying the temperature for the next iteration by the constant factor each time. This factor is computed as folows :

$$C = e^{\ln(Tend/Tstart)/(n-1)}$$

where,

T_{end}	The final temperature value,
T_{start}	The initial temperature value,
n	The number of temperature steps.

3 Results and Discussion

The river Nile is the main source of water in Egypt for irrigation and daily usage. It is quite important to predict its inflow from August to September every year. This data is used to plan the consumption of water in Egypt. The system has been applied to the Nile Inflow data at Dongla station (the entrance of Lake Nasser) to predict one season of data inflow using the recorded measured data from 1985 to 1993, and the best identified models are shown in Table 1. The results of the forecasting system are shown in Figure 1 and 2 for the off-line mode and on-line mode receptively. The results of applying statistical model selected and tuned in the conventional way are shown if Figure 3. The absolute mean forecast percentage error *(AMPFE)* is used to compare between the predicted results of the proposed implemented system GBFS, ANNs (Feed Forward network trained by the back propagation and simulated annealing algorithms), GBFS augmented by a simulated annealing algorithm and the conventional statistical systems and the results are shown Table 2.

Table 1. Parameters of the best identified models.

Model Type	**Model Parameters**
PAR Model	$\tilde{\tilde{y}}_t = 0.59\,\tilde{y}_{t-52} - 0.49\,\tilde{y}_{t-53} + 0.915\,\tilde{y}_{t-1} + 0.219\,\tilde{y}_{t-2} - 0.219\,\tilde{y}_{t-3}$
PARMA Model	$\tilde{\tilde{y}}_t = 0.3\ \tilde{y}_{t-52} + 0.075\ \tilde{y}_{t-53} + 0.014\ \tilde{y}_{t-54} + 0.48\,\tilde{e}_{t-52} + 0.45\,\tilde{e}_{t-53} + 0.45\,\tilde{e}_{t-54} + 0.9\ \tilde{y}_{t-1} - 0.28\ y_{t-2} + \tilde{e}_t - 0.32\,\tilde{e}_{t-1} - 0.16\,\tilde{e}_{t-2}$

Table 2. Error comparison between various algorithms.

Models	**AMPFE**
Off-line PAR	22.3%
Off-line PARMA	24.1%
On-line PAR	6.5%
On-line PARMA	12.8%
ANN	11%
On-line PAR (with S.A.)	5.1%
On-line PARMA (with S.A.)	4.1%
Statistical Method	26.6%

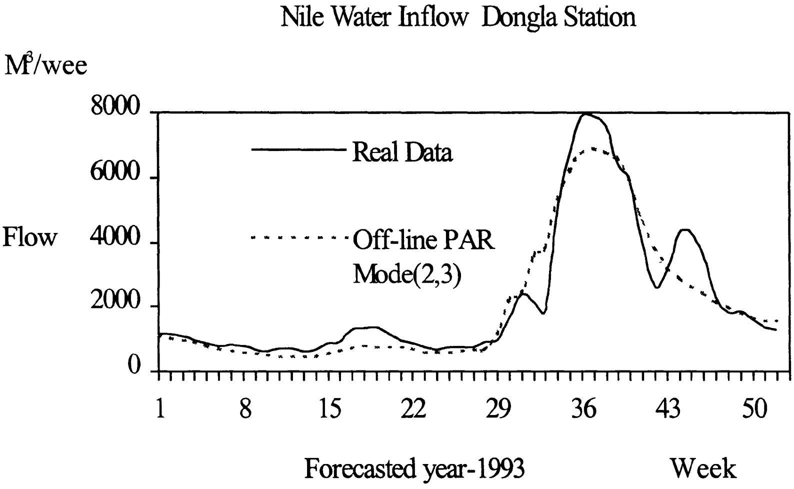


Fig. 1. The Off-line PAR model Forecasts for 52-Week Ahead using Genetic Algorithms

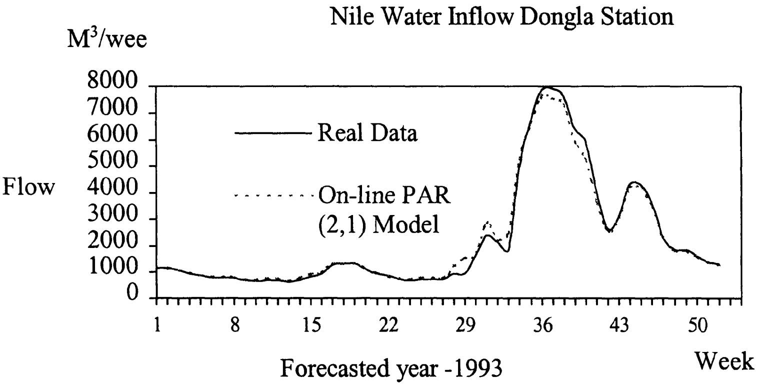


Fig. 2. The On-line PAR model Forecasts for 1-Week Ahead using Genetic Algorithms

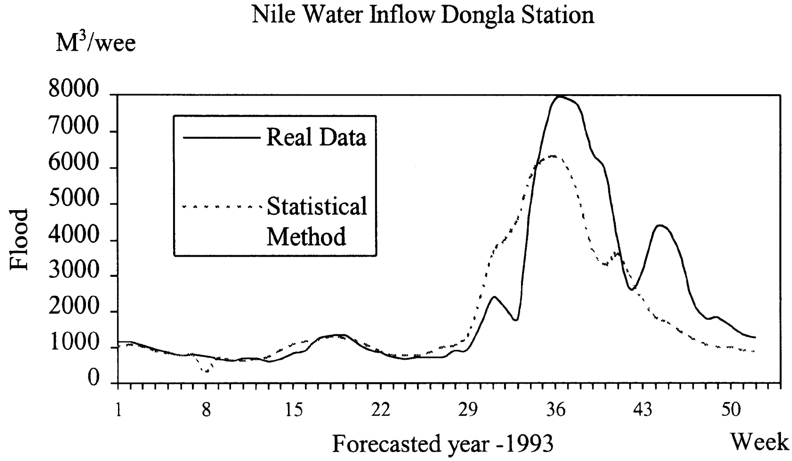


Fig. 3. The Off-line PARMA model Forecasts for 52-Week Ahead Conventional Method

The obtained results show that all the models developed in the proposed system were able to reduce the *AMPFE* in all cases as compared to the conventionally selected and tuned models. Also, the on-line mode error is much lower than the off-line-mode as new observations are treated as part of the data set in the next time step, thus avoiding the accumulation of error as the forecasting process proceeds. It should be noted that the ANN outperformed -in terms of the AMPFE- the conventional model and the GBFS without simulated annealing.

4 Conclusion

In this paper, a genetic based system for time series prediction has been presented and compared with other conventional and AI based techniques. The proposed system models have been applied to real data to forecast the Nile river water inflow and used GA as the search technique for selecting the best model and its parameters. The performance of the the models developed in the proposed system showed good results as compared to other models.

5 References

1. Box G., E., P., and Jenkins, G., M., (1976). *Time Series Analysis: Forecasting and Control.* Holden-Day, Inc.,.

2. Douglas C. Montgomery, Lynwood A. Johnson, and John S. Gardener, (1992). *Forecasting & Time Series Analysis.* John Wiley & Sons.

3. Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA.

4. Georgakakos A., P., and Yao, H., (1994). *A Routing Model For The White Nile*. Technical Report, April. Georgia Institute of Technology, Atlanta.

5. Tong, H. (1995). *Non-linear Time Series a Dynamic System Approach*. Oxford University Press Inc.

6. Michael, L. P. (1992) "Towards A more Comprehensive Theory of Learning in Computers" Doctoral dissertation, Department of Computer and Information Science, University of California, Santa Cruz.

7. Michalewicz, Z. (1994) "Genetic Algorithms + Data Structures = Evolution Programs" Springer-Verlag Berlin Heidelberg.

8. Sales, J. D., Delleur, J. W., Yevjevich, V., and Lane, W. L. (1980) "Applied Modeling of Hydraulic Time Series" Water Resources Publication , Colorado State University.

9. Zurada, J. M., (1992) "Introduction to Artificial Neural Systems" West Publishing Company.

Author Index

Lecture Notes in Artificial Intelligence (LNAI)

Vol. 1249: W. McCune (Ed.), Automated Deduction – CADE-14. Proceedings, 1997. XIV, 462 pages. 1997.

Vol. 1257: D. Lukose, H. Delugach, M. Keeler, L. Searle, J. Sowa (Eds.), Conceptual Structures: Fulfilling Peirce's Dream. Proceedings, 1997. XII, 621 pages. 1997.

Vol. 1263: J. Komorowski, J. Zytkow (Eds.), Principles of Data Mining and Knowledge Discovery. Proceedings, 1997. IX, 397 pages. 1997.

Vol. 1266: D.B. Leake, E. Plaza (Eds.), Case-Based Reasoning Research and Development. Proceedings, 1997. XIII, 648 pages. 1997.

Vol. 1265: J. Dix, U. Furbach, A. Nerode (Eds.), Logic Programming and Nonmonotonic Reasoning. Proceedings, 1997. X, 453 pages. 1997.

Vol. 1285: X. Jao, J.-H. Kim, T. Furuhashi (Eds.), Simulated Evolution and Learning. Proceedings, 1996. VIII, 231 pages. 1997.

Vol. 1286: C. Zhang, D. Lukose (Eds.), Multi-Agent Systems. Proceedings, 1996. VII, 195 pages. 1997.

Vol. 1297: N. Lavrač, S. Džeroski (Eds.), Inductive Logic Programming. Proceedings, 1997. VIII, 309 pages. 1997.

Vol. 1299: M.T. Pazienza (Ed.), Information Extraction. Proceedings, 1997. IX, 213 pages. 1997.

Vol. 1303: G. Brewka, C. Habel, B. Nebel (Eds.), KI-97: Advances in Artificial Intelligence. Proceedings, 1997. XI, 413 pages. 1997.

Vol. 1307: R. Kompe, Prosody in Speech Understanding Systems. XIX, 357 pages. 1997.

Vol. 1314: S. Muggleton (Ed.), Inductive Logic Programming. Proceedings, 1996. VIII, 397 pages. 1997.

Vol. 1316: M.Li, A. Maruoka (Eds.), Algorithmic Learning Theory. Proceedings, 1997. XI, 461 pages. 1997.

Vol. 1317: M. Leman (Ed.), Music, Gestalt, and Computing. IX, 524 pages. 1997.

Vol. 1319: E. Plaza, R. Benjamins (Eds.), Knowledge Acquisition, Modelling and Management. Proceedings, 1997. XI, 389 pages. 1997.

Vol. 1321: M. Lenzerini (Ed.), AI*IA 97: Advances in Artificial Intelligence. Proceedings, 1997. XII, 459 pages. 1997.

Vol. 1323: E. Costa, A. Cardoso (Eds.), Progress in Artificial Intelligence. Proceedings, 1997. XIV, 393 pages. 1997.

Vol. 1325: Z.W. Raś, A. Skowron (Eds.), Foundations of Intelligent Systems. Proceedings, 1997. XI, 630 pages. 1997.

Vol. 1328: C. Retoré (Ed.), Logical Aspects of Computational Linguistics. Proceedings, 1996. VIII, 435 pages. 1997.

Vol. 1342: A. Sattar (Ed.), Advanced Topics in Artificial Intelligence. Proceedings, 1997. XVIII, 516 pages. 1997.

Vol. 1348: S. Steel, R. Alami (Eds.), Recent Advances in AI Planning. Proceedings, 1997, IX, 454 pages. 1997.

Vol. 1359: G. Antoniou, A.K. Ghose, M. Truszczyński (Eds.), Learning and Reasoning with Complex Representations. Proceedings, 1996. X, 283 pages. 1998.

Vol. 1360: D. Wang (Ed.), Automated Deduction in Geometry. Proceedings, 1996. VII, 235 pages. 1998.

Vol. 1365: M.P. Singh, A. Rao, M.J. Wooldridge (Eds.), Intelligent Agents IV. Proceedings, 1997. XII, 351 pages. 1998.

Vol. 1371: I. Wachsmuth, M. Fröhlich (Eds.), Gesture and Sign Language in Human-Computer Interaction. Proceedings, 1997. XI, 309 pages. 1998.

Vol. 1374: H. Bunt, R.-J. Beun, T. Borghuis (Eds.), Multimodal Human-Computer Communication. VIII, 345 pages. 1998.

Vol. 1387: C. Lee Giles, M. Gori (Eds.), Adaptive Processing of Sequences and Data Structures. Proceedings, 1997. XII, 434 pages. 1998.

Vol. 1394: X. Wu, R. Kotagiri, K.B. Korb (Eds.), Research and Development in Knowledge Discovery and Data Mining. Proceedings, 1998. XVI, 424 pages. 1998.

Vol. 1395: H. Kitano (Ed.), RoboCup-97: Robot Soccer World Cup I. XIV, 520 pages. 1998.

Vol. 1397: H. de Swart (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. Proceedings, 1998. X, 325 pages. 1998.

Vol. 1398: C. Nédellec, C. Rouveirol (Eds.), Machine Learning: ECML-98. Proceedings, 1998. XII, 420 pages. 1998.

Vol. 1400: M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard, S. Wess (Eds.), Case-Based Reasoning Technology. XVIII, 405 pages. 1998.

Vol. 1404: C. Freksa, C. Habel. K.F. Wender (Eds.), Spatial Cognition. VIII, 491 pages. 1998.

Vol. 1409: T. Schaub, The Automation of Reasoning with Incomplete Information. XI, 159 pages. 1998.

Vol. 1415: J. Mira, A.P. del Pobil, M. Ali (Eds.), Methodology and Tools in Knowledge-Based Systems. Vol. I. Proceedings, 1998. XXIV, 887 pages. 1998.

Vol. 1416: A.P. del Pobil, J. Mira, M. Ali (Eds.), Tasks and Methods in Applied Artificial Intelligence. Vol. II. Proceedings, 1998. XXIII, 943 pages. 1998.

Vol. 1418: R. Mercer, E. Neufeldt (Eds.), Advances in Artificial Intelligence. Proceedings, 1998. XII, 467 pages. 1998.

Lecture Notes in Computer Science

Vol. 1381: C. Hankin (Ed.), Programming Languages and Systems. Proceedings, 1998. X, 283 pages. 1998.

Vol. 1382: E. Astesiano (Ed.), Fundamental Approaches to Software Engineering. Proceedings, 1998. XII, 331 pages. 1998.

Vol. 1383: K. Koskimies (Ed.), Compiler Construction. Proceedings, 1998. X, 309 pages. 1998.

Vol. 1384: B. Steffen (Ed.), Tools and Algorithms for the Construction and Analysis of Systems. Proceedings, 1998. XIII, 457 pages. 1998.

Vol. 1385: T. Margaria, B. Steffen, R. Rückert, J. Posegga (Eds.), Services and Visualization. Proceedings, 1997/1998. XII, 323 pages. 1998.

Vol. 1386: T.A. Henzinger, S. Sastry (Eds.), Hybrid Systems: Computation and Control. Proceedings, 1998. VIII, 417 pages. 1998.

Vol. 1387: C. Lee Giles, M. Gori (Eds.), Adaptive Processing of Sequences and Data Structures. Proceedings, 1997. XII, 434 pages. 1998. (Subseries LNAI).

Vol. 1388: J. Rolim (Ed.), Parallel and Distributed Processing. Proceedings, 1998. XVII, 1168 pages. 1998.

Vol. 1389: K. Tombre, A.K. Chhabra (Eds.), Graphics Recognition. Proceedings, 1997. XII, 421 pages. 1998.

Vol. 1390: C. Scheideler, Universal Routing Strategies for Interconnection Networks. XVII, 234 pages. 1998.

Vol. 1391: W. Banzhaf, R. Poli, M. Schoenauer, T.C. Fogarty (Eds.), Genetic Programming. Proceedings, 1998. X, 232 pages. 1998.

Vol. 1392: A. Barth, M. Breu, A. Endres, A. de Kemp (Eds.), Digital Libraries in Computer Science: The MeDoc Approach. VIII, 239 pages. 1998.

Vol. 1393: D. Bert (Ed.), B'98: Recent Advances in the Development and Use of the B Method. Proceedings, 1998. VIII, 313 pages. 1998.

Vol. 1394: X. Wu. R. Kotagiri, K.B. Korb (Eds.), Research and Development in Knowledge Discovery and Data Mining. Proceedings, 1998. XVI, 424 pages. 1998. (Subseries LNAI).

Vol. 1395: H. Kitano (Ed.), RoboCup-97: Robot Soccer World Cup I. XIV, 520 pages. 1998. (Subseries LNAI).

Vol. 1396: G. Davida, M. Mambo, E. Okamoto (Eds.), Information Security. Proceedings, 1997. XII, 357 pages. 1998.

Vol. 1397: H. de Swart (Ed.), Automated Reasoning with Analytic Tableaux and Related Methods. Proceedings, 1998. X, 325 pages. 1998. (Subseries LNAI).

Vol. 1398: C. Nédellec, C. Rouveirol (Eds.), Machine Learning: ECML-98. Proceedings, 1998. XII, 420 pages. 1998. (Subseries LNAI).

Vol. 1399: O. Etzion, S. Jajodia, S. Sripada (Eds.), Temporal Databases: Research and Practice. X, 429 pages. 1998.

Vol. 1400: M. Lenz, B. Bartsch-Spörl, H.-D. Burkhard, S. Wess (Eds.), Case-Based Reasoning Technology. XVIII, 405 pages. 1998. (Subseries LNAI).

Vol. 1401: P. Sloot, M. Bubak, B. Hertzberger (Eds.), High-Performance Computing and Networking. Proceedings, 1998. XX, 1309 pages. 1998.

Vol. 1402: W. Lamersdorf, M. Merz (Eds.), Trends in Distributed Systems for Electronic Commerce. Proceedings, 1998. XII, 255 pages. 1998.

Vol. 1403: K. Nyberg (Ed.), Advances in Cryptology – EUROCRYPT '98. Proceedings, 1998. X, 607 pages. 1998.

Vol. 1404: C. Freksa, C. Habel. K.F. Wender (Eds.), Spatial Cognition. VIII, 491 pages. 1998. (Subseries LNAI).

Vol. 1409: T. Schaub, The Automation of Reasoning with Incomplete Information. XI, 159 pages. 1998. (Subseries LNAI).

Vol. 1411: L. Asplund (Ed.), Reliable Software Technologies – Ada-Europe. Proceedings, 1998. XI, 297 pages. 1998.

Vol. 1413: B. Pernici, C. Thanos (Eds.), Advanced Information Systems Engineering. Proceedings, 1998. X, 423 pages. 1998.

Vol. 1414: M. Nielsen, W. Thomas (Eds.), Computer Science Logic. Proceedings, 1997. VIII, 511 pages. 1998.

Vol. 1415: J. Mira, A.P. del Pobil, M.Ali (Eds.), Methodology and Tools in Knowledge-Based Systems. Vol. I. Proceedings, 1998. XXIV, 887 pages. 1998. (Subseries LNAI).

Vol. 1416: A.P. del Pobil, J. Mira, M.Ali (Eds.), Tasks and Methods in Applied Artificial Intelligence. Vol.II. Proceedings, 1998. XXIII, 943 pages. 1998. (Subseries LNAI).

Vol. 1418: R. Mercer, E. Neufeldt (Eds.), Advances in Artificial Intelligence. Proceedings, 1998. XII, 467 pages. 1998. (Subseries LNAI).

Vol. 1422: J. Jeuring (Ed.), Mathematics of Program Construction. Proceedings, 1998. X, 383 pages. 1998.

Vol. 1425: D. Hutchison, R. Schäfer (Eds.), Multimedia Applications, Services and Techniques – ECMAST'98. Proceedings, 1998. XVI, 531 pages. 1998.

Vol. 1427: A.J. Hu, M.Y. Vardi (Eds.), Computer Aided Verification. Proceedings, 1998. IX, 552 pages. 1998.

Vol. 1430: S. Trigila, A. Mullery, M. Campolargo, H. Vanderstraeten, M. Mampaey (Eds.), Intelligence in Services and Networks: Technology for Ubiquitous Telecom Services. Proceedings, 1998. XII, 550 pages. 1998.